NUMERICAL STUDY OF ENHANCED OIL RECOVERY FROM POROUS FORMATIONS

by

ATHONU CHATTERJEE

DEPARTMENT OF MECHANICAL ENGINEERING

INDIAN INSTITUTE OF TECHNOLOGY KANPUR

December, 1993

NUMERICAL STUDY OF ENHANCED OIL RECOVERY FROM POROUS FORMATIONS

A Thesis Submitted

in Partial Fulfilment of the Requirements

for the Degree of

MASTER OF TECHNOLOGY

by

ATHONU CHATTERJEE

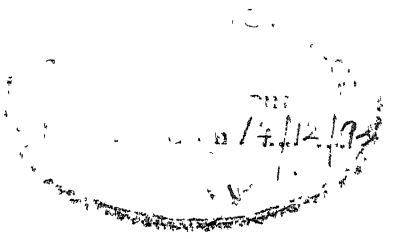
to the

DEPARTMENT OF MECHANICAL ENGINNERING

INDIAN INSTITUTE OF TECHNOLOGY KANPUR

December, 1993

ME-1993-M-CHA-NUM



CERTIFICATE

It is certified that the work contained in the thesis entitled " **Numerical study of enhanced oil recovery from porous formations**" , by Mr. Athonu Chatterjee, has been carried out under my supervision and that this work has not been submitted elsewhere for a degree.

K Muralidhar

Signature of Supervisor

Date: 14 Dec, 1993

Name: Dr. K. Muralidhar

Department: Mechanical Engg.

I. I. T. Kanpur

ACKNOWLEDGEMENT

It was my good fortune to have worked under Dr. K. Muralidhar, who has provided me his unstinted support to carry out the research work reported in this thesis. I am indebted to him for many fruitful discussions and perspicacious guidance.

CONTENTS

NOMENCLATURE

C	Tortuosity factor.
H	Consistency index in Herschel-Bulkley model, Ns^n/m.
k	Absolute permeability of the formation, m^2.
k_{eff}	Absolute permeability of the formation in the presence of shear thinning behaviour of oil, m^2.
k_n	Absolute permeability of the formation in the presence of Newtonian behaviour of oil, m^2.
$k_{ro,w}$	Relative permeability of oil and water respectively.
K	Thermal conductivity, W/m^oC.
L	Length of the domain, m.
n	Power law index.
P_c	Capillary pressure between oil and water phases, N/m^2.
$P_{o,w}$	Phase pressures of oil and water respectively, N/m^2.
$P_{1,2}$	Absolute pressures on injection and exit planes, N/m^2.
P_I	Absolute pressure at oil-water interface, N/m^2.
ppv	Percentage pore volume of oil recovery.
$S_{o,w}$	Saturation of oil and water in the porous region.
t, Δt	Time and time step, s.
T	Temperature, ^oC.
T_f	Temperature of rock formation, ^oC.
u	Darcy velocity, m/s.
U	Composite velocity in the energy equation, m/s.
U_{front}	Velocity of the water front, m/s.
x, y	Cartesian coordinates, m.
Δx, Δy	Grid size along x and y axis repectively, m.

Greek Symbols

β	Expansivity, $^oC^{-1}$.
ξ	Compressibility, Pa^{-1}.
μ	Dynamic viscosity of the fluid phase, N m/s.
ε	Porosity.

Superscript

n	Current time step.

Subscript

i,j	Node indices along x and y axis respectively.
n	Special case of a Newtonian fluid.

ABSTRACT

A numerical study of multi-phase flow in an oil reservoir is presented in this work. Two phase flow of oil and water is mathematically modelled and the effect of surfactants in enhancing oil recovery is examined. Surfactants are surface active chemicals that reduce considerably the capillary forces in the fluid. Since the bulk of resistance to flow in unsaturated porous media comes from capillary forces, the use of surfactants can be expected to be quite effective in improving oil recovery. The governing equations are set up in terms of oil and water pressures. The pressure equations are seen to be coupled and non-linear. A fully implicit finite difference scheme has been used to solve the governing partial differential equations. Non-linearity is accounted for using Picard iterations within each time step. Both Newtonian as well as shear thinning behaviour of oil are considered in the analysis.

In the present work a domain decomposition technique has been developed for solving the governing equations. Domain decomposition is a useful method for solving problems over very large domains or regions having a complex geometry. An attractive feature of this technique is the possibility of parallelizing the computer code suitable for computers with a parallel architecture. In the present work a completely parallel code has been developed for the oil recovery problem using the domain decomposition technique. Results do show a great potential for this technique while solving oil recovery problems over a field scale.

It is now widely recognized that there could be a critical world oil shortage in near future. In this situation a possibility of significantly increasing the recovery of oil from existing reservoirs assumes special importance. It is of course known that that a large proportion of the oil still remains unrecovered. This is so because until very recently there was no real economic incentive to develop enhanced oil recovery (EOR) technologies. Things have however changed fast and we are now beginning to see a sense of urgency in developing these techniques. Particularly in our country where oil production is negligible and EOR techniques are still a vague concept, much needs to be done in this direction. It is the objective of this work to consider the EOR techniques from numerical point of view and develop faster and more reliable numerical codes for reservoir simulation.

Before the advent of enhanced oil recovery (EOR) techniques, conventional methods were used for oil recovery. These conventional methods fall in the following two categories:

1) Primary recovery: In this case the pressure in the reservoir is high enough to force the fluid out without any pumping effort. This accounts for 15%-20% of the extraction.

2) Secondary recovery: Here water, air or steam injection drives are used to maintain the reservoir pressure. Some of the oil is physically displaced towards the well by the high pressure injection. Oil extraction of around 50% is achieved at this stage.

A large fraction of the world's oil reservoirs remains unrecoverable by the conventional recovery methods. The three major reasons for this drawback are the following. Owing to the tortous nature of the microscopic pores in which oil is trapped only a portion of oil reserve can be contacted by the displacing fluid. Secondly, an irreducable limit of oil saturation exists in the porous region and not all of the oil can be displaced from the pores by the high pressure displacing fluid. Thirdly, certain oils are too viscous to move at sufficient rates to support an economically feasible operation. Keeping these facts in mind, many advanced techniques collectively known as enhanced oil recovery (EOR) techniques, have been developed over the past two decades.

Field experiments have confirmed that these are more efficient than the conventional techniques when the oil saturation drops below a certain value. There are several EOR techniques in current use. All of them attempt to overcome one or more of the three factors mentioned above. For example, thermal recovery methods (*Boberg (1988)*) aim at reducing the oil viscosity by raising its temperature thus improving the displacement efficiency of oil. Polymer flooding (*Boberg (1988), Shah (1977)*) is used to increase the volume of the reservoir contacted by the displacing fluid.

Lately, surfactant flooding has become popular among the EOR techniques because of its high efficiency and relative ease with which it can implemented in practice. Surfactants are surface active chemicals that reduce the interfacial tension between oil and water considerably. A surfactant solution in water injected at an appropriate pressure mobilizes the trapped oil to form a flowing oil bank by overcoming the capillary forces that are predominant in unsaturated porous regions. However, surfactant flooding is not devoid of problems. To start with, surfactants are very high cost chemicals that are not found in abundance. Further, a significant amount of laboratory work is needed to tailor the surfactant system for each reservoir.

In the present work we numerically study the effect of surfactants on oil recovery. Oil recovery with surfactants dissolved in water is compared with oil recovery without surfactants. An appreciable increase in oil recovery in the presence of surfactant is seen. An ideal case is also considered in which the capillary resistance is zero, corresponding to a 100% effective surfactant. Both Newtonian and non-Newtonian behaviour of oil are considered throughout the analysis. For the non-Newtonian behaviour, oil is modelled as a power law fluid exhibiting shear thinning behaviour. The Herschel-Bulkley model (*Aziz (1986)*) is used to modify accordingly the governing Darcy's law. Oil recovery is seen to increase further when shear thinning behaviour is included in the model.

Reservoir simulation is one of the most challenging problems in engineering (*Ewing (1983), Jim (1983)*). Owing to variety of contributing factors like the inherent complexity of the domain, complex nature of the constitutive relationships and the

instabilty of the oil-water interface the governing partial differential equations are highly non-linear and coupled. While analytical solutions are difficult to obtain, mundane numerical methods exhibit instability and consequent divergence.

In this work a pressure based formulation (*Ewing (1983)*) is used. The phase pressures are solved for using the control volume finite difference scheme. The scheme is fully implicit and hence is expected to be unconditionally stable. As mentioned before, the oil-water interface is inherently unstable and explicit schemes require excessively small time steps. The presence of mass transfer phenomenon such as adsorption and desorption of surfactants from water to the solid matrix and the vice versa is neglected in the present work. The temperature of the formation is taken to be a constant since the study is focussed on the effect of surfactants on oil recovery.

Computational time is an important aspect to be kept in mind while solving reservoir problems. Because of high degree of non-linearity of the governing equations and large size of reservoirs, reservoir simulation can be computationally intensive even on the most powerful computers available at this time. The advent of parallel processing promises new generation of computers that can deliver an order of magnitude improvement in computing speeds. A parallel algorithm is accordingly needed that is compatible with the parallel architecture. In this context, domain decomposition techniques (*Glowinski et al (1983)*) are very suitable for reservoir simulation. In domain decomposition the physical domain is split into sub-domains. The governing equations are then solved independently in each sub-domain. The individual solutions must be assembled carefully to get a converged solution of the original problem on the complete physical domain. Domain decomposition can rapidly make the simulation parallelizable since calculations in each sub-domain can be carried out independently on a proceessor. They can also handle large problems on sequential machines more efficiently.

A large body of works on domain decomposition have appeared in the literature in the past few years. Many of them originate from the Schwarz method (*Glowinski et al (1983)*) for solving elliptic problems. This method is originally sequential in

nature. Lately it has been modified in view of its good parallelization properties *(Glowinski et al (1983))*. *Glowinski et al (1983)*, in their pioneering work have developed a number of efficient parallelizable domain decomposition algorithms and these are currently in wide use. *Perng and Street (1990)* have applied domain decomposition technique to solve Navier-Strokesequations in complex domains which are otherwise difficult to solve without using coordinate transformation. Parralelizable domain decomposition codes applied to transient problems are however scant in literature. Sequential domain decomposition codes applied to steady problems are available in abundance (*Perng (1990)*).

In the present work we have developed a parallel code to solve the transient oil recovery problem. Uzawa's algorithm (*Glowinski (1983)*, *Yagawa (1991)*) has been used for interface treatment in conjunction with an implicit formulation. The overall algorithm has the desired parallel and temporal properties. Results are in excellent agreement with those of a full domain simulation with no observed deterioration in the sequential CPU time.

The thesis is organized along the following lines:

1) Mathematical model and formulation of the oil recovery problem.
2) Developement of domain decomposition technique.
3) Results of simulation of enhanced oil recovery.
4) Scope for future work.

We start this section with the derivation of the equations governing oil and water pressures in two phase flow through a porous medium.

2.1 ASSUMPTIONS

The following assumptions have been made:

1) The two fluid phases namely, oil and water flow simultaneously.

2) The fluids are immiscible i.e. there is no mass transfer between them.

3) Gravity term is neglected in the momentum equation. It is reasonable because the time frames are short and as a result the gravity override effect will not show up.

4) The porous formation is incompressible and porosity is a constant.

5) Relative permeabilities of phases and capillary pressure are only a function of saturation. The fluid viscosities depend on temperature only.

6) There is no net mass source in the flow field.

2.2 PHYSICAL PRINCIPLES

Following are the basic equations that govern flow through a homogeneous, isotropic porous region. These are used to derive the final pressure equations. In this derivation oil and water are assumed to exhibit Newtonian behaviour. Modifications arising from non-Newtonian behaviour of oil are presented later.

a) Mass balance

Mass balance equations for the oil and water phases are the following.

oil: $$\frac{\partial (\varepsilon S_o \rho_o)}{\partial t} + \underset{\sim}{\nabla}. \underset{\sim}{u}_o \rho_o = q_o \qquad (2.1)$$

water: $$\frac{\partial (\varepsilon S_w \rho_w)}{\partial t} + \underset{\sim}{\nabla}. \underset{\sim}{u}_w \rho_w = q_w \qquad (2.2)$$

Since the formation is incompressible, ε, the porosity is a constant. The mass source terms, q_o and q_w are taken to be zero in this work. ρ_o and ρ_w are the phase densities. S_o and S_w are oil and water saturation repectively. u_o and u_w are the Darcy velocities of oil and water respectively. We assume that no gas phase is present in the formation and so,

$$S_o + S_w = 1 \qquad (2.3)$$

b) Momentum equation:

The momentum equation is taken in the form of a generalized Darcy's law as follows.

oil: $$\underset{\sim}{u}_o = - \frac{k\, k_{ro}}{\mu_o} (\underset{\sim}{\nabla} P_o) \qquad (2.4)$$

water: $$\underset{\sim}{u}_w = - \frac{k\, k_{rw}}{\mu_w} (\underset{\sim}{\nabla} P_w) \qquad (2.5)$$

Here, k is the absolute permeability of the porous formation, k_{ro} and k_{rw} are the relative permeabilities of oil and water respectively, μ_o and μ_w are the phase viscosities and P_o and P_w are the phase pressures.

c) Equations of state:

The quantities ξ and β defined below are taken to be specified.

compressibility: $$\xi_o = \frac{1}{\rho_o} \frac{\partial \rho_o}{\partial P_o}\Bigg|_T \qquad \xi_w = \frac{1}{\rho_w} \frac{\partial \rho_w}{\partial P_w}\Bigg|_T \qquad (2.6)$$

Volumetric thermal expansion coefficient:

$$\beta_o = -\frac{1}{\rho_o}\frac{\partial \rho_o}{\partial T}\bigg|_{P_o} \qquad (2.7)$$

$$\beta_w = -\frac{1}{\rho_w}\frac{\partial \rho_w}{\partial T}\bigg|_{P_w}$$

d) Constitutive relations:

The formation properties given below are also taken to be available from laboratory or field experiments.

Capillary pressure:

$$P_o - P_w = P_c(S_w) \qquad (2.8)$$

Relative permeability:

$$k_{ro} = k_{ro}(S_w) \qquad (2.9)$$

$$k_{rw} = k_{rw}(S_w) \qquad (2.10)$$

Here subscripts o and w stand for oil and water respectively. T in the definition of β is the local (cell-averaged) temperature.

2.3 GOVERNING DIFFERENTIAL EQUATION

We incorporate the equations of state and the constitutive relations into the mass balance equation for oil. Using Equations 2.1, 2.3 and 2.7, we get after simplification the following oil pressure equation:

$$\varepsilon S_o \rho_o \left[\xi_o \frac{\partial P_o}{\partial t} - \beta_o \frac{\partial T}{\partial t} \right] + \varepsilon \rho_o \frac{\partial S_o}{\partial t} = \underset{\sim}{\nabla} \cdot \frac{k\, k_{ro} \rho_o}{\mu_o} \underset{\sim}{\nabla} P_o \qquad (2.11)$$

Similarly for water pressure, we get

$$\varepsilon S_w \rho_w \left[\xi_w \frac{\partial P_w}{\partial t} - \beta_w \frac{\partial T}{\partial t} \right] + \varepsilon \rho_w \frac{\partial S_w}{\partial t} = \underset{\sim}{\nabla} \cdot \frac{k\, k_{rw} \rho_w}{\mu_w} \underset{\sim}{\nabla} P_w \quad (2.12)$$

Equations 2.11 and 2.12 may be written as,

$$-S_o \beta_o \frac{\partial T}{\partial t} + S_o \xi_o \frac{\partial P_o}{\partial t} + \frac{\partial S_o}{\partial t} = \frac{1}{\varepsilon \rho_o} \left[\underset{\sim}{\nabla} \cdot \frac{k\, k_{ro} \rho_o}{\mu_o} \underset{\sim}{\nabla} P_o \right] \quad (2.13)$$

$$-S_w \beta_w \frac{\partial T}{\partial t} + S_w \xi_w \frac{\partial P_w}{\partial t} + \frac{\partial S_w}{\partial t} = \frac{1}{\varepsilon \rho_w} \left[\underset{\sim}{\nabla} \cdot \frac{k\, k_{rw} \rho_w}{\mu_w} \underset{\sim}{\nabla} P_w \right] \quad (2.14)$$

At this stage we introduce the capillary pressure, P_c, since k_{ro}, k_{rw}, S_o and S_w are unique functions of P_c. Using Equation 2.8 and the relation,

$$\frac{\partial S_w}{\partial t} = \frac{dS_w}{dP_c} \frac{\partial P_c}{\partial t}$$

Equations 2.13 and 2.14 reduce to the following:

$$\text{oil:}\quad -S_o\beta_o \frac{\partial T}{\partial t} + S_o\xi_o \frac{\partial P_o}{\partial t} - \frac{dS_w}{dP_c}\left[\frac{\partial P_o}{\partial t} - \frac{\partial P_w}{\partial t}\right] = \frac{1}{\rho_o}\underset{\sim}{\nabla}\cdot\left[\frac{k\, k_{ro}\rho_o}{\varepsilon\,\mu_o}\right]\underset{\sim}{\nabla} P_o \qquad (2.15)$$

$$\text{water:}\quad -S_w\beta_w \frac{\partial T}{\partial t} + S_w\xi_w \frac{\partial P_w}{\partial t} + \frac{dS_w}{dP_c}\left[\frac{\partial P_o}{\partial t} - \frac{\partial P_w}{\partial t}\right] = \frac{1}{\rho_w}\underset{\sim}{\nabla}\cdot\left[\frac{k\, k_{rw}\rho_w}{\varepsilon\,\mu_w}\right]\underset{\sim}{\nabla} P_w \qquad (2.16)$$

Equations 2.15 and 2.16 are the final form of the pressure equations used in the present work for numerical simulation. Once P_o and P_w are known other quantities can be determined from the constitutive relations. The energy equation in its final form is given as,

$$\frac{\partial T}{\partial t} + \underset{\sim}{U}\cdot\underset{\sim}{\nabla} T = \frac{K}{\sigma_T}\nabla^2 T \qquad (2.17)$$

where $\underset{\sim}{U} = \frac{1}{\sigma_T}\sum_i \rho_i C_i \underset{\sim}{u}_i$, $\sigma_T = \varepsilon\left\{\sum_i S_i\rho_i C_i\right\} + (1-\varepsilon)\,(\rho C)_{formation}$

in which C_i is the specific heat of phase i (= o, w) and K is the thermal conductivity of the formation. Other symbols hold the same meaning as mentioned before.

It is necessary to justify the use of pressure formulation at this stage. Oil recovery problem can also be formulated in terms of water saturation. This approach can however lead to difficulties in numerical computation since the oil-water interface can be quite sharp. In contrast to this the pressure field must be continuous everywhere. This results in considerable improvement in the accuracy of the numerical simulation.

2.4 NON - NEWTONIAN BEHAVIOUR

We look at the non-Newtonian behaviour of oil. Field experiments show that oil can be modelled as a power-law fluid exhibiting shear-thinning behaviour. Darcy's law, as it appears in Equations 2.3 and 2.4 may be written in its general form as

$$\underset{\sim}{u}_i = -\frac{k(\underset{\sim}{\nabla} P_i)\, k_{ri}(P_c)}{\mu_i} (\underset{\sim}{\nabla} P_i - G) \qquad (2.18)$$

where the subscript i refers individually to the oil and water phases. G is the thresold pressure gradient below which there is no flow. It has been taken to be zero in this study. We see that k, the absolute permeability, depends on the local pressure gradient. This dependance is absent when the fluids exhibit Newtonian behaviour. The shear-thinning behaviour of oil is well-described by the Herschel-Bulkley model (*Aziz (1986)*). In this model the appropriate form of k in Equation 2.18 for the oil phase is given as,

$$k_{eff} = \mu_o \left[\frac{k_n}{\mu_{eff}} \right]^{1/n} \left| \underset{\sim}{\nabla} P_o \right|^{1/n-1} \qquad (2.19)$$

where k_n is the absolute permeability of the formation in the absence of non-Newtonian behaviour and μ_{eff} is given as,

$$\mu_{eff} = \frac{H}{4} \left(3 + \frac{1}{n} \right)^n (8\, C\, \varepsilon\, k_n)^{(1-n)/2} \qquad (2.20)$$

In the above equation H is the consistency index of the Herschel-Bulkley model and n is the flow behaviour index equivalent to the index in a power-law fluid. C is the tortuosity factor of the formation. Based on the data presented by (*Aziz (1986)*), C is taken as 25/12 in the present study. Equations 2.19 and 2.20 reduce to the Newtonian case of $\mu_{eff} = H = \mu_o$ and $k_{eff} = k_n$ for n=1. The surfactant solution can also exhibit non-Newtonian behaviour. In this study the surfactant solution is however assumed to be Newtonian.

2.5 GEOMETRIC MODEL

The geometry of the physical region used in the present study is shown in Figure 2.1. The flow is two dimensional and the rectangular domain has a nominal size of 4m x 0.4m. Thus, if the domain has a length of Xm and breadth of Ym then

$$X = 4m \quad \text{and} \quad Y = 0.4m$$

Further, in a Cartesian x-y plane x=0 and x=X are the inflow and outflow planes respectively. On these planes constant pressures are specified. The upper and the lower edges of the domain i.e. y=0 and y=0.4 are impermeable. We solve for the oil and water pressures separately. The difference between the two pressures is the capillary pressure P_c, and is a measure of the surface tension. The capillary pressure prevents equalization of the phase pressures and hence, $P_c = P_o - P_w$ is non-zero. Typical values of the parameters appearing in Equations 2.6 and 2.7, the equations of state, are taken from *Boberg (1988)* and summarized in Table 2.1. The constitutive relations for an unsaturated porous medium namely, $k_{ri} = k_{ri}(S_w)$ and $P_c = P_c(S_w)$ are shown in Figures 2.2a and 2.2b. Since $\Sigma S_i = 1$, the functional relationship can be shown in terms of one phase saturation alone. In the present study results have been obtained in terms of water saturation. An increase in S_w with time clearly shows oil displacement. Since the effect of surfactants is to reduce the capillary forces and mobilize the oil bank, the capillary pressure P_c decreases and the relative permeabilities, k_{ri}, increase in its presence. This effect is clearly shown in Figures 2.2a and 2.2b. Curves a, b and c refer to the relationships in the absence of surfactant, in the presence of surfactant and in the presence of an ideal surfctant respectively. Owing to lack of availability of real field data, these constitutive relations show the general effect of surfactants. They are in tune with the relations presented in *Barenblatt et al (1986).*

2.6 INITIAL CONDITION AND BOUNDARY CONDITIONS

The pressure equations, Equations 2.15 and 2.16, are numerically solved on a rectangular domain, Figure 2.1, subject to a quiscient initial condition, impermeable confining walls on which $\partial P_i/\partial y =$

0, where i refers individually to the oil phase and water phase, and specified oil and water pressures on the inflow and outflow planes. This model predicts the movement of the water front from the inflow plane into the oil-rich porous zone. This increase in water saturation is a measure of the amount of oil displaced. Oil displacement is measured in terms of the percentage pore volume (ppv). It is defined as, $ppv = 100 \times \frac{\text{total volume of oil produced}}{\text{total pore volume}}$

The use of Dirichlet boundary condition on the outflow plane is justified only if the location of this plane is far away from the region effected by fluid injection. In the present study oil recovery (ppv) is computed over a distance of 2m and the outflow plane is located at a distance of 4m from the inflow plane. For the time period of three hours considered in the present simulation the water moves a distance of approximately 1m. Numerical experiments show that oil recovery is insensitive to the location of the outflow plane for the time period considered here.

2.7 CASE OF IDEAL SURFACTANT

We now dicuss the limiting case of an ideal surfactant which when added to the injected fluid eliminates surface tension altogether. Hence the relative permeabilty and capillary pressure reach their limiting values of unity and zero respectively as shown in Figures 2.2a and 2.2b. A water front moves through the porous medium under these conditions displacing oil completely. We assume that the water saturation is at its maximum value ahead of the front. Similarly, beyond the front water saturation is minimum. Since the effect of surfactant is totally negated flow resistance arises from the phase viscosities alone. For a domain of length L the average oil and water velocities are equal and equal to the speed of the front movement. For the Newtonian case this speed is estimated from the principle that the front speed computed from the oil side be equal to that computed from the water side.
Hence,

$$U_{front} = \frac{k\,(P_1 - P_I)}{\mu_w x} = \frac{k\,(P_I - P_2)}{\mu_o (L - x)} \qquad (2.21)$$

Here P_1, P_I and P_2 are the injection plane, interface and exit

plane pressures respectively. x indicates the depth penetrated by the water-oil interface. Eliminating P_I in Equation 2.21 the expression for the front speed is obtained as,

$$U_{front} = \frac{k}{\mu_w \frac{x}{L} + \mu_o \left(1 - \frac{x}{L} \right)} \times \frac{(P_1 - P_2)}{L} \qquad (2.22)$$

At t = 0, x = 0 and at any time t we update x as, $x_{new} = x_{old} + U_{front} \Delta t_o$. Here U_{front} is calculated from Equation 2.21 using the old value of x. Oil recovery is then measured in terms of ppv as, $ppv = \frac{100x}{L}$.

When the shear-thinning behaviour of oil is taken into consideration, a closed form solution for U does not exist. In terms of P_1, P_2 and P_I as well as the power-law index n, the modified form of Darcy's law , Equation 2.18, can be written as,

$$U_{front} = \left[\frac{k_n}{\mu_{eff}} \right]^{\frac{1}{n}} \times \left[\frac{P_I - P_2}{L - x} \right]^{\frac{1}{n}} = - \frac{k_n}{\mu_w} \left[\frac{P_I - P_1}{x} \right] \qquad (2.23)$$

For each x Equation 2.23 is solved for P_I iteratively. With this value of P_I the interface velocity U_{front} is calculated as,

$$U_{front} = \left[\frac{k_n}{\mu_{eff}} \right]^{\frac{1}{n}} \times \left[\frac{P_I - P_2}{L - x} \right]^{\frac{1}{n}} \qquad (2.24)$$

The same procedure as given after Equation 2.20 is adopted to obtain x at any time t and finally the ppv.

Table 2.1 : Properties Used in Numerical Calculation

Absolute Permeability, K = 132 Darcies

Porosity, ε = 0.375

Compressibility: oil, $\xi_o = 0.03447\ Pa^{-1}$, water, $\xi_w = 0.02137\ Pa^{-1}$

In-situ formation pressure = 1.31 MPa

Injection pressure = 1.793 MPa

Dynamic Viscosity (Pa-s)

T, ^oC	20	40	60	80
Oil	4.12	2.37	0.61	0.10
Water	1.03×10^{-3}	6.6×10^{-4}	4.73×10^{-4}	3.67×10^{-4}

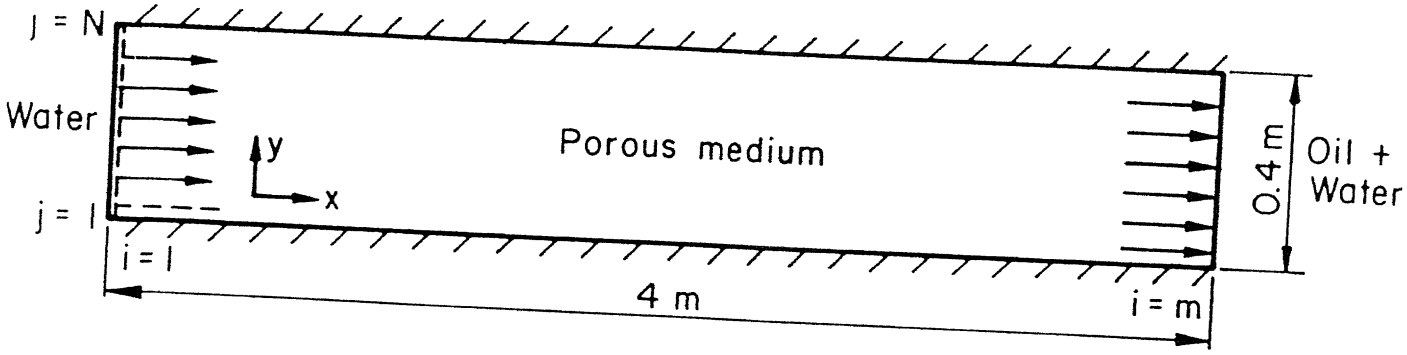


Figure 2.1 Flow configuration and coordinate system.

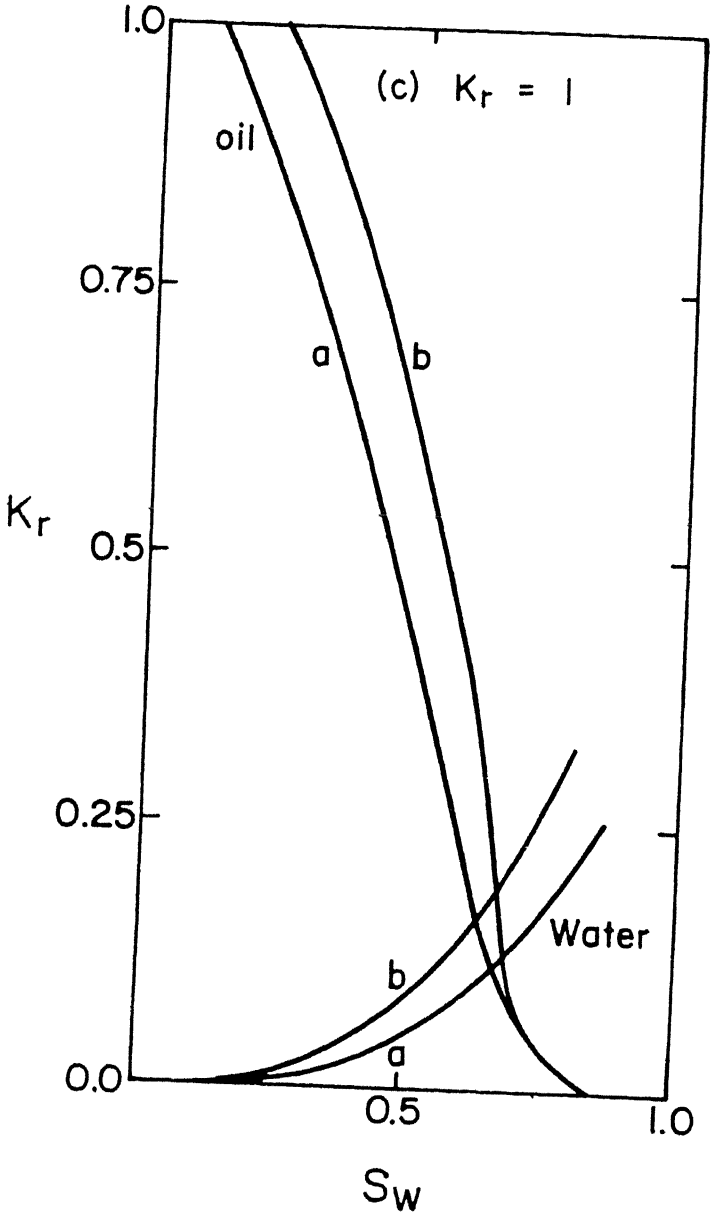


Figure 2.2a Relative permeability as function of saturation.

a) Without surfactant b) With surfactant c) Ideal case

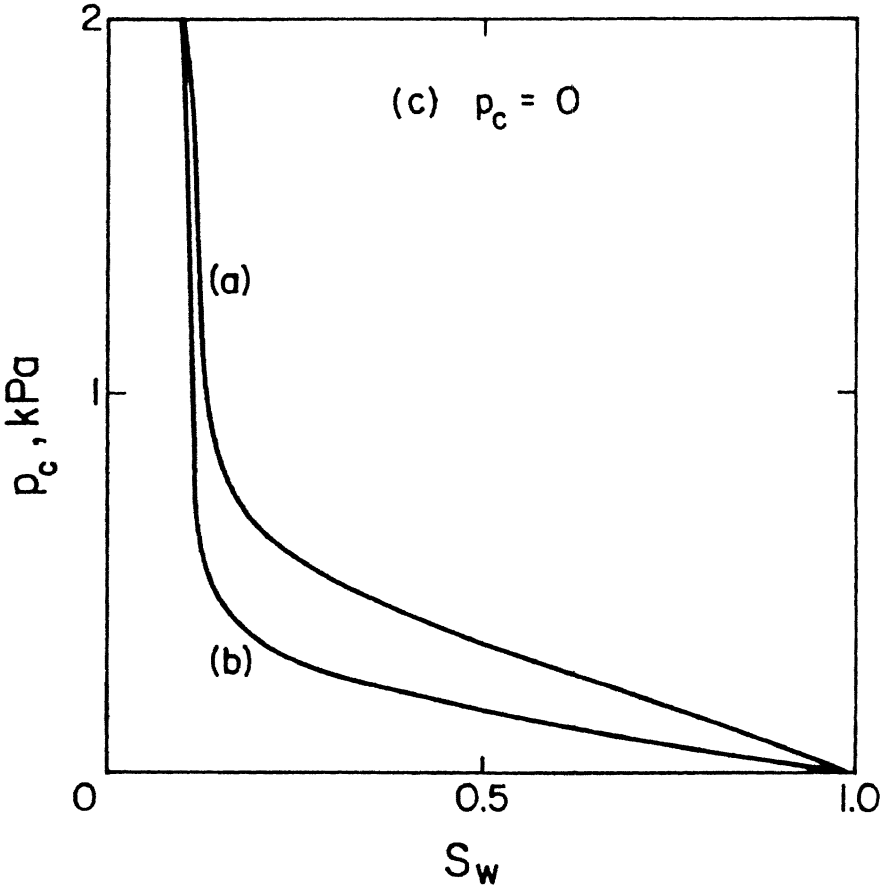


Figure 2.2b Capillary pressure as function of saturation.

a) Without surfactant b) With surfactant c) Ideal case

3.1 OPENING REMARKS

Practical engineering problems inevitably involve complex geometries and large domains. Computation of three dimensional viscous flows, high speed gas flows, turbulent flows with separation and enhanced oil recovery require enormous amount of computing time and memory space. In many case the computations cannot be implemented on existing computers at all. The current approach towards complex problems is to improve the numerical algorithm on one hand and computer performance on the other hand. With the advent of parallel computers there exists a great potential in increasing computer speeds as well as memory. Parallel computers require parallel algorithm to exploit the distributed nature of the processor architecture. Domain decomposition techniques are the simplest of parallel algorithms that can convert a sequential algorithm to one suitable for parallel computing. Besides the ability to parallelize, other specific advantages of domain decomposition are given below.

1) It can generate smaller matrices that require less memory and are rapidly invertible when iterative techniques are used.

2) Complex assemblage of components of a system can be analyzed systematically.

3) Complex phenemena can be localized by assigning separate sub-domains to them.

3.2 MODEL PROBLEM

We demonstrate domain decomposition techniques first through a simple example. Consider a linear governing differential equation of the type

$$\Delta u = f \quad \text{in } \Omega \tag{3.1}$$

$$\text{and} \quad u = f \text{ on } \partial\Omega$$

where f and g are known functions. Here Ω is the physical region and $\partial\Omega$ is its boundary as shown in Figure 3.1. Δ is the Laplacian operator.

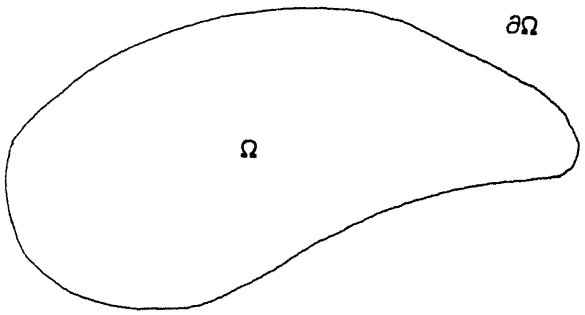


Figure 3.1

The basic theme of domain decomposition is that a large domain is divided into many sub-domains that are easier to handle and are linked at the interfaces. These sub-domains may be overlapping or non-overlapping. We first consider the case of non-overlapping sub-domains.

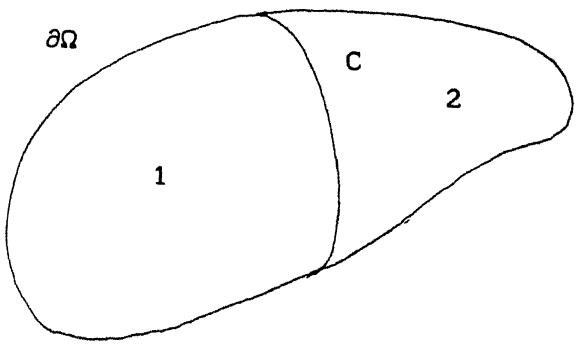


Figure 3.2

Figure 3.2 shows a division of the full domain into two sub-domains, 1 and 2. The interface between the two sub-domains is denoted C. We seek a solution of Equation 3.1 in the original large domain. The problem in region 1 is solved numerically with an arbitrary Dirichlet boundary condition on C. Let this solution be u_1^1, u being the original dependent variable. This is followed

by numerical solution of u in region 2 with a boundary condition on C which is derived from u_1^1. The global solution u and its derivative should be continous at the interface. Hence the most obvious choice of the boundary condition on C for region 2 would be the normal derivative $\partial u_1^1/\partial n$. Here n denotes the normal at C exterior to region 1. With this boundary condition u can be solved in region 2. Let this solution be u_2^1. This marks the end of one complete sweep of the original domain. Thus, after one complete sweep of the original domain we have solved the following equations:

In region 1,

$$\Delta u_1 = f \qquad (3.2)$$
$$u_1 = g \text{ on } \partial\Omega \text{ and } u_1 = \lambda \text{ on } C$$

In region 2,

$$\Delta u_2 = f \qquad (3.3)$$
$$u_2 = g \text{ on } \partial\Omega \text{ and } \frac{\partial u_2}{\partial n_2} = -\left.\frac{\partial u_1}{\partial n_1}\right|_C \text{ on } C$$

where λ is the Dirichlet boundary condition described before and n_1 and n_2 are the normals at C exterior to regions 1 and 2 respectively.

The problem in region 1 is solved again with a new boundary condition on C. This may be the value of u_2^1 on C or some combination of u_2^1 and u_1^1 on C. Let the solution be denoted u_1^2. The procedure continues in this fashion where the problem in regions 1 and 2 are solved alternately until convergence in the global solution is achieved. This convergence criterion will satisfy the conditions of continuity of u and its derivative at the interface. This method described above is known as the alternating Dirchilet-Neumann (D-N) method because Dirchilet and Neumann boundary conditions are applied alternately on the interface.

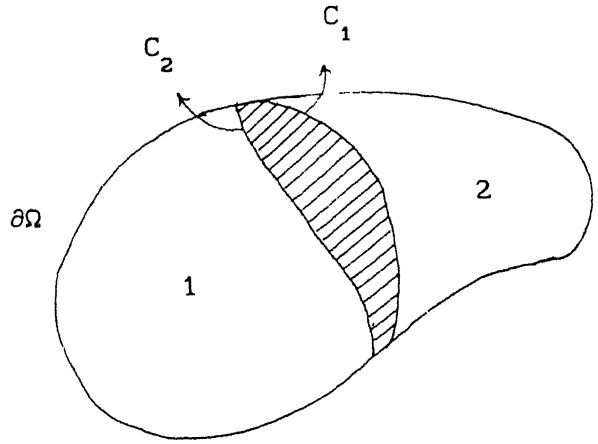


Figure 3.3

We now look at domain decomposition with overlapping sub-domains. The method described below is also known as the Schwarz-alternating method. Figure 3.3 shows two sub-domains 1 and 2, with a slight overlap that is shaded. C_1 is the interface of region 1 and C_2 is that of region 2. We consider the solution of Equation 3.1 again in the full domain Ω. The sequence of calculations is as follows. The problem in region 1 is solved numerically with an arbitrary boundary condition on C_1. Let the solution be u_1^1. Then the problem in region 2 is solved with a boundary condition on C_2 that is already obtained in region 1 i.e. u_1^1 on C_2, since C_2 falls in region 1. Let the solution be denoted u_2^1. Subsequent iterations are generated as follows.

The problem in region 1 is solved, with a new boundary condition on C_1 that may be the value of u_2^1 on C_1 or a linear combination of u_1^1 and u_2^1 on C_1. This is equivalent to the use of under-relaxation factor for the numerical scheme. The iterations continue until convergence is achieved. The convergence criterion should ensure that after I iterations, u_2^I and u_1^I are sufficiently close. In a finite difference scheme if the overlap is such that there are a few nodes lying in it then the convergence criteria should be such that after I iterations, u_2^I and u_1^I at these nodes are sufficiently close. Only then do we get a globally converged solution for u.

One major advantage of Schwarz-alternating scheme is that it can do away with Neumann boundary condition altogether. This

however requires that the overlap be chosen judiciously. In a finite difference scheme if the overlap is only of a grid size then by solely using Dirichlet boundary conditions on C_1 and C_2 we can ensure the continuity of the derivatives at the overlap region. This is a major advantage because Dirichlet boundary conditions are easier to handle and are computationally faster when iterations are involved, as in the non-linear problems.

The two domain decomposition methods described above are sequential in nature. Their efficiency depends on how the iterations are handled at the sub-domain interfaces. Consider the alternating D-N method as an example. After one complete sweep of the domain, the use of arithmatic mean of u_1^I and u_2^I as the boundary condition on C for u_1^{I+1} can lead to a considerable improvement in overall convergence rate. The alternating Schwarz method also shows similar improvement. Thus, interfaces are the most crucial zones and they should be handled with care and intelligence.

The ultimate utility of domain decomposition techniques lies in their ability to permit parallel computing. Domain decomposition techniques have been rediscovered primarily for this feature. We now describe a domain decomposition algorithm that permits parallelization. It is known as Uzawa's algorithm. It employs non-overlapping sub-domains. For definitiveness the problem in Equation 3.1 is reconsidered. For a globally converged solution we require both u and its derivative to be continuous at the interface C (Figure 3.2). Keeping this in mind we start with a guess value of the derivatives, $\left.\frac{\partial u}{\partial n_1}\right|_C$ and $-\left.\frac{\partial u}{\partial n_2}\right|_C$, where n_1 and n_2 are the normals at C exterior to regions 1 and 2 respectively. Let this guess be λ^1. Using λ^1 as the boundary condition on C we solve the governing differential equation in regions 1 and 2. Let the solutions be u_1^1 and u_2^1. Let y_1^1 and y_2^1 be the values of u_1^1 and u_2^1 repectively on C. Then λ^1 is updated as

$$\lambda^2 = \lambda^1 + \rho(y_2^1 - y_1^1) \qquad (3.4)$$

where ρ is a suitable constant that lies between 0 and 1.
Using λ^2 as the next guess we repeat the earlier steps. Thus the overall algorithm has the following structure:
1) Assume λ^1, where λ^1 is the normal derivative at the interface

exterior to the sub-domains.

2) Solve the governing partial differential equation in the two sub-domains, 1 and 2. i.e.

In sub-domain 1

$$\Delta u_1 = f \qquad (3.5)$$
$$u_1 = g \text{ on } \partial\Omega \text{ and } \frac{\partial u_1}{\partial n_1} = \lambda^1 \text{ on } C$$

In sub-domain 2

$$\Delta u_2 = f \qquad (3.6)$$
$$u_2 = g \text{ on } \partial\Omega \text{ and } \frac{\partial u_2}{\partial n_2} = -\lambda^1 \text{ on } C$$

where n_1 and n_2 are the normals at C exterior to the sub-domains 1 and 2 respectively.

3) Using the values of u_1^1 and u_2^1 at the interface i.e. y_1^1 and y_2^1, λ^1 is updated as in Equation 3.4. Thus after I iterations,

$$\lambda^{I+1} = \lambda^I + \rho(y_2^I - y_1^I) \qquad (3.7)$$

It can be seen that this scheme is completely parallel. For example, calculations in regions 1 and 2 can be carried out independently on two different processors. The processors return the values of y_1^I and y_2^I after each iteration. Iterations can be stopped when

$$\left| \lambda^{I+1} - \lambda^I \right| < \varepsilon \qquad (3.8)$$

where ε is the desired convergence criterion.

As mentioned before, Uzawa's algorithm can be implemented on a parallel computer. We now describe briefly how this can be implemented on a hypercube-type parallel computer(*Ganagi et al (1991)*). The hypercube architecture has been adopted by a variety of research groups all over the world. We describe the implementation of the Uzawa's algorithm on PACE, a parallel computer of the hypercube-type developed by ANURAG, a defence research lab under the ministry of defence, Government of India. PACE-8 has eight concurrent processors and similarly PACE-128 has 128 concurrent processors. Hence with PACE-8 as many as eight sub-domains can be handled.

The complete parallel computer code will have two parts. The

first part is the computation-intensive portion that deals with the numerical solution of the governing equations in each sub-domain. These calculations are performed on individual processors. The second part is the front-end-processor (FEP) which deals with keeping track of interface values, checking for their convergence and downloading the initial and boundary conditions onto the sub-domains at the beginning of each iteration. At the beginning of each time step the front-end-processing part supplies the required initial values at the interface as well as the internal points of the sub-domains. The boundary conditions are also supplied. Then the computation intensive part takes over and with the supplied values calculations are performed in the sub-domains on the respective processors in parallel. After local convergence is achieved in all the sub-domains, the front-end-processing part collects the final solutions for the sub-domains and checks the interface values for convergence. In this fashion iterations proceed and upon interface convergence the front-end-processing part moves on to the next time step.

3.3 SPECIFIC ADVANTAGES OF DOMAIN DECOMPOSITION

We list below the advantages of domain decomposition techniques in the solution of engineering problems.

1) Parallelization: As discussed above, domain decomposition has the ability to parallelize the computer code. Hence the computations are faster and the problem of memory is circumvented.

2) Complex domains: Domain decomposition techniques are much simpler to use in domains with complicated geometries. Using domain splitting the complex domain can be divided into many regular sub-domains. The governing equations can be solved easily on the regular sub-domains.

3) Large domain simulation on sequential machines:
Even in the absence of parallel computing domain decomposition can still be appreciably faster than ordinary simulations for large domains. Let us elaborate on this point. We consider a domain

decomposed problem with two sub-domains against an ordinary (full domain) simulation. For a finite difference code with a matrix size N (number of discretizedequations) the convergence rate of an iterative matrix varies as 1/N. Thus, if e is the residual error after I iterations, e is given as

$$e = K\ I^{a/N} \tag{3.9}$$

where K and a are constants. Note that the convergence rate goes to zero as N goes to infinity. Hence iterative inversion of large matrices is slow unless an extremely good initial guess is available.

Let E be the desired level of residual error and I_1 be the number of iterations required. Equation 3.9 can be written as,

$$\log E = \frac{a}{N} \log I_1 + \log K \tag{3.10}$$

and so

$$I_1 = \left(\frac{E}{K} \right)^{N/a}$$

Let us now split the original physical domain into two equal sub-domains. The order of matrix arising from finite differences in each sub-domain becomes N/2. The number of iterations required to the reach the same level of residual error , E, would now be

$$I_2 = \left(\frac{E}{K} \right)^{N/a} \tag{3.11}$$

Let D be the number of iterations required at the interface. Keeping in mind that there are two sub-domains, the total number of iterations for the domain decomposed problem would be

$$I_s = 2DI_2 = 2D \left(\frac{E}{K} \right)^{N/2a} \tag{3.12}$$

Thus,

$$\frac{I_s}{I_1} = 2D \left(\frac{E}{K} \right)^{-N/2a} \tag{3.13}$$

A plot of (I_s/I_1) versus N is shown in Figure 3.4 .

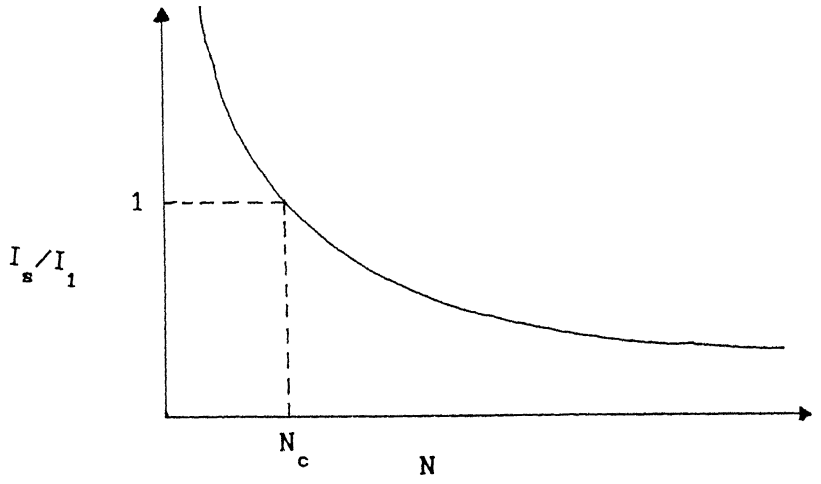


Figure 3.4

Note that above a critical value of N, $(I_s/I_1) < 1$, i.e. the domain decomposed problem requires lesser number of iterations than the full domain problem.

3.4 DOMAIN DECOMPOSITION FOR NON - LINEAR PROBLEMS

Throughout the earlier sections we have assumed that the governing problem is linear (Equation 3.1). In case of a non-linear problem the interface treatment must be modified. Let us take up an example to illustrate this point. Consider the linear heat conduction problem in the same domain as in Figure 3.2:

$$K \underset{\sim}{\nabla}^2 T = \rho c_p \frac{\partial T}{\partial t} \tag{3.14}$$

where K, ρ and c_p are constants.

In Equation 3.14 the boundary conditions to be matched at the interfaces of the sub-domains would be the temperature and the temperature derivative, $\frac{\partial T}{\partial n}$. This is both physically and mathematically conceivable. Consider the non-linear problem,

$$\underset{\sim}{\nabla}. \; K \underset{\sim}{\nabla} T = \rho c_p \frac{\partial T}{\partial t} \tag{3.15}$$

where K is taken as a function of temperature. The physically conceivable Neumann boundary condition to be matched at the interface would be not the derivatives of T but the fluxes at the

interface , $-K(T)\frac{\partial T}{\partial n}$. In case of the linear problem this boils down to the temperature derivatives alone since K is not a function of temperature. Thus for the non-linear problem, global convergence requires

$$-K(T_1^I)\left.\frac{\partial T_1^I}{\partial n_1}\right|_C = K(T_2^I)\left.\frac{\partial T_2^I}{\partial n_2}\right|_C \qquad (3.16)$$

It should be noted that during iterations $T_1^I \neq T_2^I$. Also, in inhomogeneous media with sharp change in thermal conductivity at the interface, heat fluxes rather than the temperature gradients should be matched.

In the present chapter we develop the formulae applicable for numerical simulation of oil recovery. We first discuss the full domain simulation and then proceed to the formulation with domain decomposition. The governing differential equations and boundary conditions are the same as derived in Chapter 2.

4.1 FULL DOMAIN SIMULATION

A central difference scheme is used for the numerical solution of governing differential equations. The physical domain is shown in Figure 4.1. The control volume is shown below, Figure 4.2.

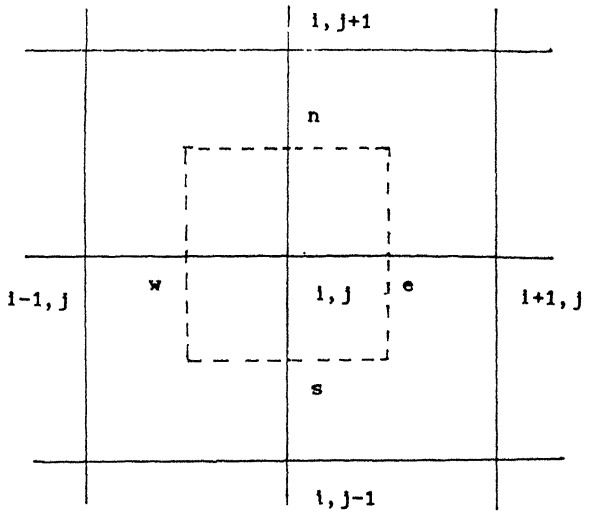


Figure 4.2

Let N and M be the number of nodes in x and y directions respectively. Δx and Δy define a Cartesian mesh and Δt is the time step. The current level is represented by n and the future time level by n+1. The scheme is fully implicit with respect to time stepping and is expected to be unconditionally stable.

The oil pressure equation, Equation 2.15, is reconsidered here. It is given as,

$$-S_o\beta_o \frac{\partial T}{\partial t} + S_o\xi_o \frac{\partial P_o}{\partial t} - \frac{dS_w}{dP_c}\left[\frac{\partial P_o}{\partial t} - \frac{\partial P_w}{\partial t}\right]$$

$$= \frac{1}{\rho_o} \underset{\sim}{\nabla}\cdot \left[\frac{k\, k_{ro}\rho_o}{\varepsilon\, \mu_o}\right] \underset{\sim}{\nabla} P_o$$

(4.1)

We now discretize this equation using finite differences. Since the formation temperature is taken to be constant in this study, the first term in the left hand side namely, $-S_o\beta_o\frac{\partial T}{\partial t}$, is dropped. Let

$$\theta = \frac{K(\underset{\sim}{\nabla}P_o)\, K_{ro}(S_w)\, \rho_o(T)}{\varepsilon\, \mu_o(T)}$$

Then the right hand side is $\frac{1}{\rho_o} \underset{\sim}{\nabla}\cdot \theta \underset{\sim}{\nabla} P_o$ and is equal to

$$\frac{1}{\rho_{oi,j}} \frac{\left[\theta \frac{\partial P_o}{\partial x}\right]_e - \left[\theta \frac{\partial P_o}{\partial x}\right]_w}{\Delta x} + \frac{1}{\rho_{oi,j}} \frac{\left[\theta \frac{\partial P_o}{\partial y}\right]_n - \left[\theta \frac{\partial P_o}{\partial y}\right]_s}{\Delta y}$$

$$= \frac{1}{\rho_{oi,j}} \frac{\theta_e \left[\frac{P_{oi+1,j} - P_{oi,j}}{\Delta x}\right]^{n+1} - \theta_w \left[\frac{P_{oi,j} - P_{oi-1,j}}{\Delta x}\right]^{n+1}}{\Delta x} +$$

$$\frac{1}{\rho_{oi,j}} \frac{\theta_n \left[\frac{P_{oi,j+1} - P_{oi,j}}{\Delta y}\right]^{n+1} - \theta_s \left[\frac{P_{oi,j} - P_{oi,j-1}}{\Delta y}\right]^{n+1}}{\Delta y}$$

Here θ_e, θ_w, θ_n and θ_s are evaluated as harmonic averages of their values at nodes that lie on either side, Figure 4.2. Hence

$$\theta_e = \frac{2\,\theta_{i,j}\theta_{i+1,j}}{\theta_{i,j} + \theta_{i+1,j}}$$

Similarly,

$$\theta_w = \text{harmonic average } (\theta_{i,j} , \theta_{i-1,j})$$

$$\theta_n = \text{harmonic average } (\theta_{i,j} , \theta_{i,j+1})$$

$$\theta_s = \text{harmonic average } (\theta_{i,j} , \theta_{i,j-1})$$

The left hand side is discretized as,

$$\left(S_o \xi_o - \frac{dS_w}{dP_c} \right)_{i,j} \left[\frac{P^{n+1}_{oi,j} - P^{n}_{oi,j}}{\Delta t} \right] + \left[\frac{dS_w}{dP_c} \right] \left[\frac{P^{n+1}_{oi,j} - P^{n}_{oi,j}}{\Delta t} \right]$$

The full equation in discretized form may be written as,

$$\left[A_o P_{oi+1,j} + B_o P_{oi,j} + C_o P_{wi,j} + D_o P_{oi-1,j} + E_o P_{oi,j+1} + F_o P_{oi,j-1} \right]^{n+1}$$

$$= G_o \qquad (4.2)$$

The water pressure equation, Equation 2.16, can be discretized similarly to yield,

$$\left[A_w P_{wi+1,j} + B_w P_{wi,j} + C_w P_{oi,j} + D_w P_{wi-1,j} + E_w P_{wi,j+1} + F_w P_{wi,j-1} \right]^{n+1}$$

$$= G_w \qquad (4.3)$$

The coefficients $A_o - G_o$, $A_w - G_w$ are given in Appendix A2.

The discretized boundary conditions are as follows:

Specified inflow plane and outflow plane pressures:

$$\left.\begin{aligned} P_o(1,j) &= P_{o1} \\ P_w(1,j) &= P_{w1} \\ P_o(N,j) &= P_{o2} \\ P_w(N,j) &= P_{w2} \end{aligned}\right] \qquad (4.4)$$

Impermeable confining walls at y = 0 and y = Y:

$$\left.\begin{aligned} P_o(i,1) - P_o(i,2) &= 0 \\ P_w(i,1) - P_w(i,2) &= 0 \\ P_o(i,M) - P_o(i,M-1) &= 0 \\ P_w(i,M) - P_w(i,M-1) &= 0 \end{aligned}\right] \qquad (4.5)$$

The pressure equations, Equations 4.2 and 4.3, are coupled and are solved simulataneously. When Equations 4.2-4.5 are written for each node, a set of 2NM non-linear algebraic equations is obtained.

Non-linearity has been primarily handled using Picard iterations within each time step. In this scheme the coefficients, A_o-G_o, A_w-G_w, depend on P_o and P_w. As a first guess they are evaluated using the previous time step values of P_o, P_w and S_w. Iterations continue within each time step until convergence in P_o and P_w are achieved. The matrix structure arising from Equations 4.2-4.3 is shown in Figure 4.3. It is banded and has an overall size of 2NM × 2NM. This matrix is inverted using an efficient sparse matrix inverter (*Duff (1980)*).

4.2 ALGORITHM

We now outline the algorithmic steps required to obtain P_o, P_w, S_w and ppv as a function of time.

1) The initial distribution of P_o, P_w and S_w at t = 0 are prescribed.

2) The coefficients, A_o-G_o, A_w-G_w of the pressure equations are computed using the previous time step values of P_o, P_w and S_w.

3) The system of pressure equations is solved to obtain new values of P_o and P_w.

4) From P_o and P_w, S_w is obtained using the constitutive relations.

5) Steps 1-4 are repeated until convergence of P_o and P_w is achieved within the time step.

6) S_w is updated and ppv is calculated using the updated S_w values.

7) Fresh computation is initiated for the next time step starting from step 2.

4.3 APPLICATION OF NEWTON - RAPHSON TECHNIQUE

A Newton-Raphson technique has also been used in the present work to solve the system of coupled pressure equations, Equations 4.2-4.5. The Newton-Raphson iteration formulae are described in Appendix A2. Owing to abrupt changes in the value of capillary pressure, P_c in the low saturation zone, Figure 4.4, calculation of the derivative, dS_w/dP_c, in this zone becomes highly susceptible to errors. Further, there are abrupt changes in the value of derivative itself. Hence Newton-Raphson iterations are not sustained and the scheme diverges in general after a few iterations. One has to employ very small time steps and small relaxation factors to sustain the iterations. Hence our study shows that Newton-Raphson iterations are not suitable for oil recovery simulation since the constitutive relations do not possess smooth features. However a truncated Newton-Raphson scheme has been successfully used in this study. Here the derivative terms for the rapidly changing properties are dropped from the elements of the Jacobian matrix. This is seen to give accurate results using the same computational time as in Picard iterations. Since there is no additional advantage using the truncated Newton-Raphson method, all results have been presented here using Picard iterations alone.

4.4 ENHANCED OIL RECOVERY SIMULATION USING DOMAIN DECOMPOSITION

We now describe the oil recovery simulation using domain decomposition. The physical domain is shown in Figure 4.1 and the initial and boundary conditions are identical to those considred in Section 4.1. Uzawa's algorithm is used for interface treatment. Hence the computer code is suitable for parallel computing. Implicit formulation is used in each sub-domain to circumvent time step limitations.

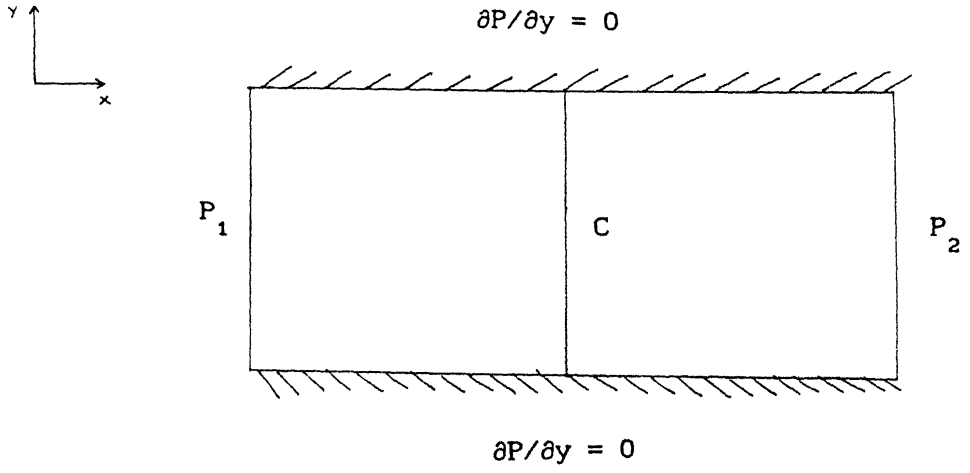


Figure 4.5

As shown in Figure 4.5, the physical domain is split into two equal sub-domains. It should be noted at this stage that Uzawa's algorithm can accomodate any number of sub-domains without any additonal complications. We limit ourselves to two sub-domain formulation to make the mathematical description more comprehensible. The interface, C, is a straight line parallel to the y-axis. This choice of interface saves a considerable amount of effort that may be required while calculating the derivatives along the outward drawn normal. Thus, for the sub-domains 1 and 2, boundary conditions on three edges are automatically prescribed from the original problem and only the interface has to be taken care of. We are solving for the phase pressures, P_o and P_w, in each sub-domain. It should be noted that the problem is non-linear. Hence the quantities to be matched at the interface are the phase pressures , P_o and P_w and the Darcy velocities of oil and water. This is in analogy with flux matching as described in chapter 3. These Darcy velocities are, Equations 2.4-2.5, $-\frac{K K_{ro}}{\mu} \frac{\partial P_o}{\partial x}$ and $\frac{K K_{rw}}{\mu} \frac{\partial P_w}{\partial x}$. Since K and μ are constants with respect to pressure for the Newtonian case, we are left with the

pressure fluxes, $-K_{ro}\frac{\partial P_o}{\partial x}$ and $K_{rw}\frac{\partial P_w}{\partial x}$, to match. Derivatives with respect to x are calculated along the outward normal of the sub-domains. We now outline the algorithm developed here:

1) At the beginning of each time step we supply the guess values of the pressure fluxes at the interface. The most judicious choice would be the fluxes calculated in the previous time step. Let us call them λ_o^1 and λ_w^1.

2) With λ_o^1 and λ_w^1 chosen, we solve the pressure equations in sub-domains 1 and 2 using λ_o^1 and λ_w^1 as the boundary conditions at the interface. A central difference scheme as described in Section 4.1 is used to solve the pressure equations in regions 1 and 2. Let the solutions be P_{o1}^1 and P_{w1}^1 in region 1 and P_{o2}^1 and P_{w2}^1 in region 2.

3) Using Uzawa's algorithm we update the fluxes as

$$\lambda_o^2 = \lambda_o^1 + \rho \left(P_{o2}^1\Big|_C - P_{o1}^1\Big|_C \right) \tag{4.6}$$

$$\lambda_w^2 = \lambda_w^1 + \rho \left(P_{w2}^1\Big|_C - P_{w1}^1\Big|_C \right) \tag{4.7}$$

A value of $\rho = 0.25$ has been found to be optimum in this study.

4) Steps 2-3 are repeated until the conditions

$$\left| \lambda_o^{I+1} - \lambda_o^I \right| < \varepsilon \text{ and } \left| \lambda_w^{I+1} - \lambda_w^I \right| < \varepsilon \tag{4.8}$$

are satisfied. Here I indicates the level of interface iteration.

5) Using the pressure values calculated in regions 1 and 2, S_w is updated and ppv is calculated.

6) We proceed to the next time step and the sequence of operations, steps 1-5, is repeated.

In Table 4.1 we compare the saturation values calculated using the full domain simulation and domain decomposition codes. The values are seen to be extremely close . Table 4.2 shows the comparison between the CPU times taken for three hour simulation using full domain simulation and domain decomposition code on a refined grid of $\Delta x = 0.0167$m. Domain decomposition can be seen to be faster even on a sequential machine. In Table 4.3 we show the effect of number of sub-domains on CPU time through a non-linear heat conduction problem using Gauss-Seidel iterations.

Table 4.1: Comparison of full domain simulation and simulation with domain decomposition for the oil recovery problem.

x in meters and t in hours. Table shows water saturation values at a formation temperature of 30°C. Newtonian behaviour of oil is considered.

A: Full domain simulation B: Domain Decomposition

x →		0.5	1	1.5	2.0	2.5	3.0	3.5	4.0
t= 1.5	A	.3295	.2353	.2232	.2011	.2000	.2000	.2000	.2000
	B	.3297	.2349	.2232	.2012	.2000	.2000	.2000	.2000
t= 3.0	A	.3656	.2760	.2532	.2325	.2198	.2055	.2000	.2000
	B	.3656	.2762	.2539	.2340	.2202	.2059	.2000	.2000

Table 4.2: Comparison of CPU times for full domain simulation and simulation with domain decomposition (two sub-domains) for three hour simulation of the oil recovery problem in the presence of surfactant.

CPU time in seconds. Temperature T in ° C.

Oil is taken to be Newtonian.

	Full domain	Domain Decomposition
T=30	900	845
T=40	920	866
T=60	933	917

Table 4.3: Comparison showing the effect of number of sub-domains on CPU time.

CPU time (in seconds) are for steady state simulation of the non-linear heat conduction problem described in chapter 5 using Gauss-Seidel iterations. N is the number of sub-domains. N = 0 indicates full domain simulation.

A: 241 nodes

B: 3001 nodes

	N=0	N=2	N=3
A	1.0	1.1	0.85
B	23.2	20.6	18.4

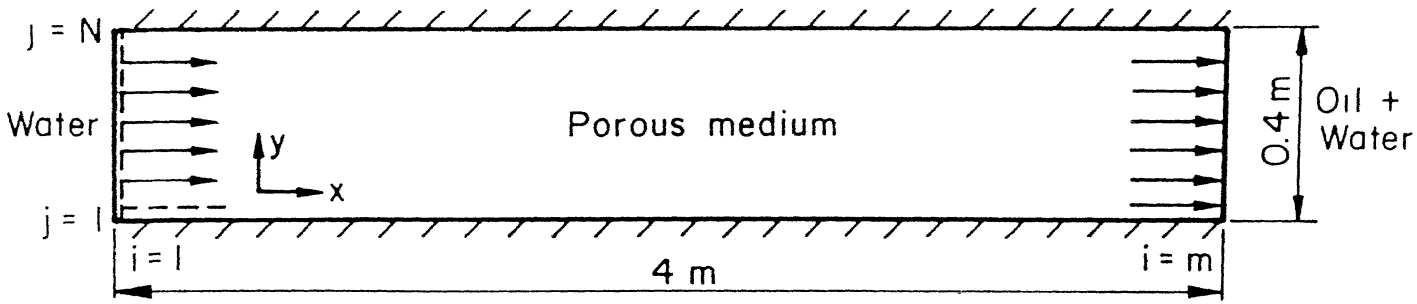


Figure 4.1 Flow configuration and coordinate system.

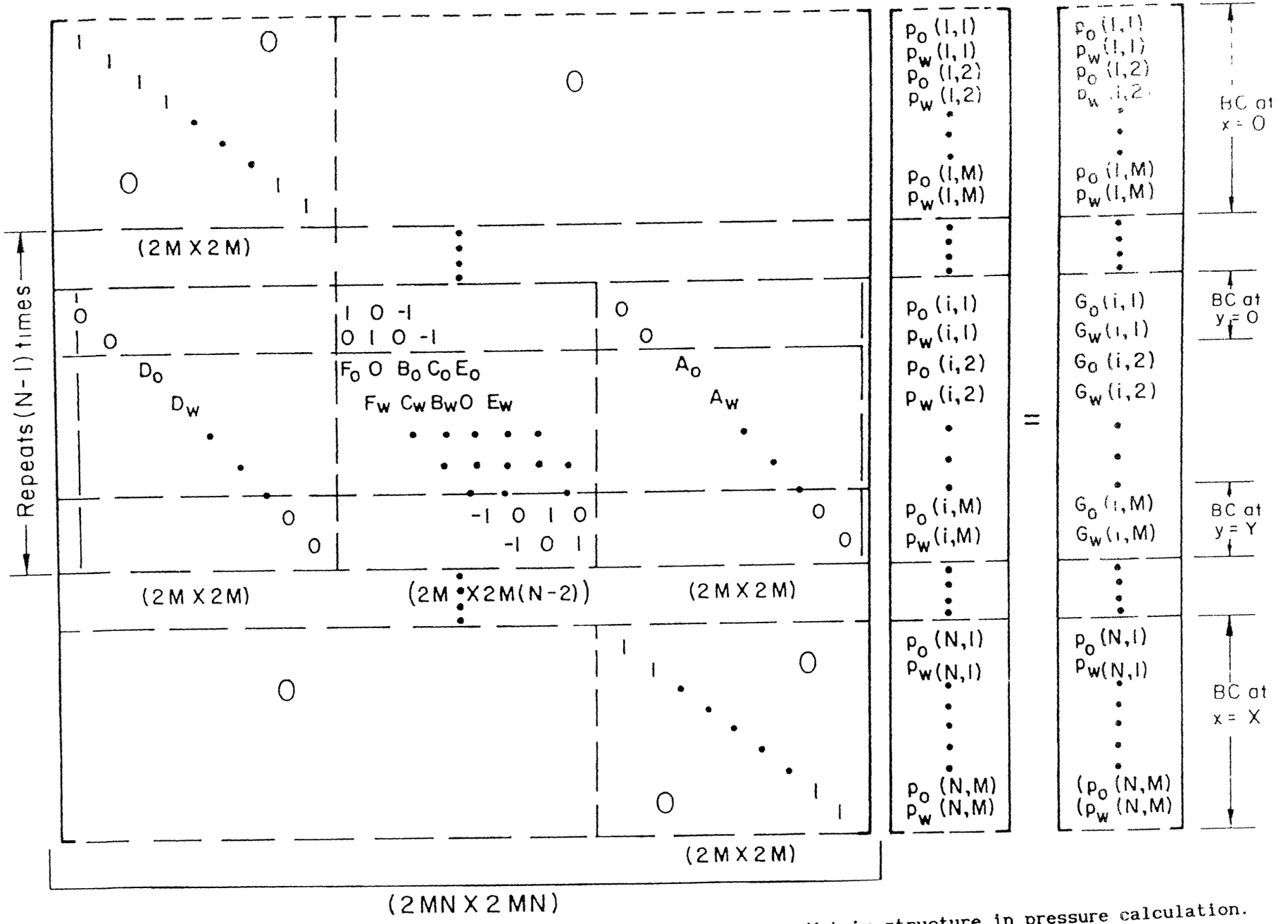


Figure 4.3 Matrix structure in pressure calculation.

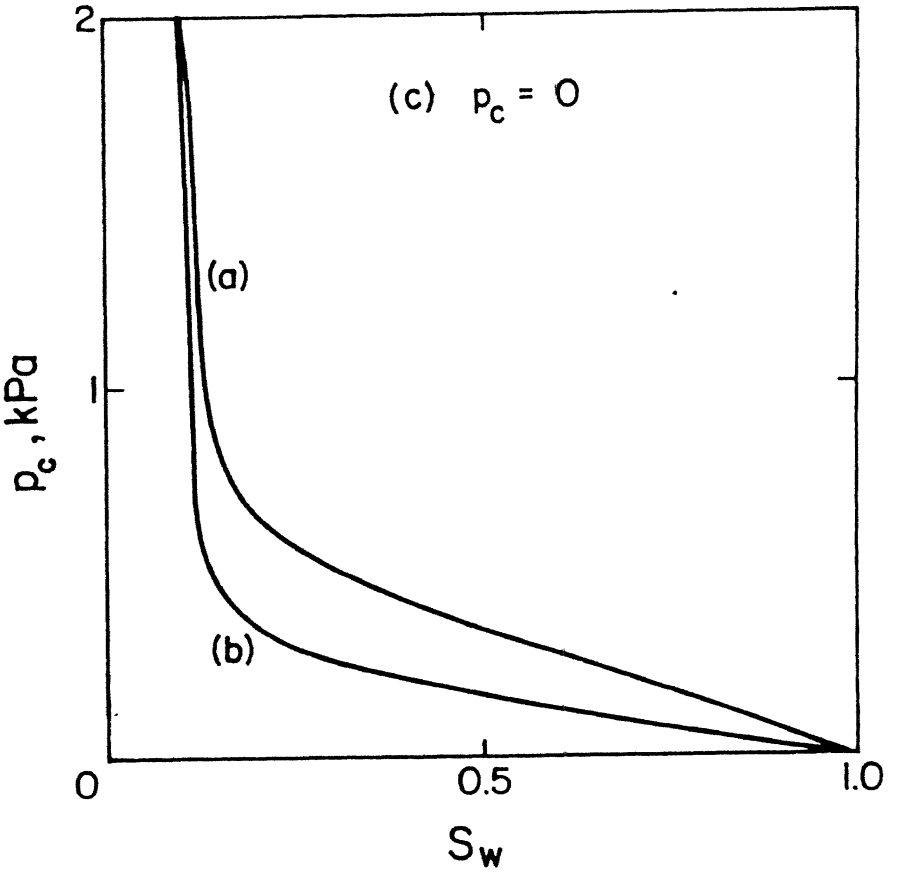


Figure 4.4 Capillary pressure as function of saturation.

a) Without surfactant b) With surfactant c) Ideal case

In the present chapter the results have been given under the following heads:

1) Effect of surfactants on oil recovery.

2) Effect of shear-thinning behaviour on oil recovery.

3) Effect of enhanced inflow plane pressure on the saturation profile and oil recovery.

5.1 DATA

Fluid properies such as oil and water and formation properties are given in Table 5.1. The constitutive relations namely, $k_r = k_r(S_w)$ and $P_c = P_c(S_w)$ are shown in Figures 5.1a and 5.1b. Oil is modelled as a Newtonian fluid (n=1) as well as a shear thinning fluid (n<1). For the study of shear thinning behaviour, modified Darcy's law, Equation 2.18, is used and the value of n is taken to be 0.6. This value of n encompasses a variety of crude oils (*Aziz (1986)*) that are generally found in oil reservoirs. The formation temperature and injection temperature of water are taken to be equal. Temperature is however used as a model parameter as oil and water viscosities are strong functions of temperature. The ppv values and saturation profiles presented in this chapter correspond to a three hour simulation.

5.2 GRID

The grid sizes used are $\Delta x = 0.05$m and $\Delta y = 0.2$m and the time step used varies from Δt=3.6 seconds for the initial time levels to Δt=36 seconds for the later ones. Further grid refinement does not show any significant changes in the numerical results and this point is demonstrated in Table 5.2. A convergence criterion of

$$100 \times \frac{\phi^{p+1} - \phi^{p}}{\phi^{p+1}} \leq 0.01 \qquad (5.1)$$

where ϕ is P_o or P_w, is used for the Picard iterations within a time step.

In order to minimize spurious oscillations, smoothened initial conditions for P_o, P_w and S_w are used. These spurious oscillations are observed for short times when initial conditions are not smooth. Initial conditions are of the form

For $0 \leq x \leq 0.4$,

$$P_o = 1.79 - 1.225x \quad \text{MPa} \tag{5.2}$$

$$S_w = 0.86 - 1.65x \tag{5.3}$$

$$P_c = P_c(S_w) \quad \text{MPa} \tag{5.4}$$

$$P_w = P_o - P_c \quad \text{MPa} \tag{5.5}$$

For $x > 0.4$,

$$P_o = 1.31 \text{ MPa} \tag{5.6}$$

$$S_w = 0.2 \tag{5.7}$$

$$P_c = P_c(S_w) \text{ MPa} \tag{5.8}$$

$$P_w = P_o - P_c \text{ MPa} \tag{5.9}$$

The computer code deveoped here is written in FORTRAN 77 and has been implemented on HP 9000. The full domain simulation code has around 600 lines. The domain decomposition code has been written in such a manner that it can be implemented on a parallel computer without major modifications. Calculations in sub-domains are performed in separate subroutines and a main program similar to the front-end-processing part described in Chapter 4 invokes them and performs the necessary operations. The main program has been written in a versatile manner so that it can accomodate as many sub-domains as necessary.

5.3 TESTING

Veracity of all the computer codes developed namely, full domain simulation with Picard iterations and Newton-Rapson iterations, domain decompostion code and the matrix inverter has been tested by solving the non-linear heat conduction problem

$$\underset{\sim}{\nabla}.\ K(T)\ \underset{\sim}{\nabla}\ T = \frac{\partial T}{\partial t} \tag{5.6}$$

With the same initial and boundary conditions as described in Appendix A1. The reference solution is taken from an independent computer program that uses Gauss-Seidel iterations for the same problem. Values for comparison are shown in Table 5.3. The analytical solution for T (one dimensional problem) at steady state for $K(T) = 1 - 0.4T$ is

$$T = \frac{5 - \sqrt{16x+9}}{2} \tag{5.7}$$

5.4 EFFECT OF SURFACTANT ON OIL RECOVERY

Results obtained by modeling oil as a Newtonian fluid are considered first. Figure 5.2 shows oil recovery in terms of ppv as a function of the formation temperature. The three cases namely, injection without surfactant, with surfactant and the ideal case are compared here. It is seen in Figure 5.2 that ppv increases with temperature. An increase in temperature lowers oil viscosity and hence the mobility of the oil-water bank. The effect of lowering surface tension in the presence of surfactants can also be seen in the figure. A dramatic increase in ppv of about seven times is seen in the presence of ideal surfactants. Even in the presence of partially effective surfactants the increase is appreciable.

Figure 5.3, compares water saturation profiles after three hours of injection with and without surfactants. It can be seen that water migration is diffuse in the case of injection without surfactants and with partially effective surfactants. The saturation curve for the latter case is always higher than for the former case showing greater mobility of the oil-water bank in the presence of surfactants. In the ideal case water moves as a front since capillary resistance is zero. Surfactants that show this degree of effectiveness remain to be synthesized at this time.

5.5 EFFECT OF SHEAR THINNING BEHAVIOUR OF OIL

Figure 5.4, delineates the effect of shear-thinning behaviour of oil. Oil recovery in the presence of shear-thinning behaviour is compared with that in the presence of Newtonian behaviour for all the three cases mentioned before. Shear-thinning behaviour is characterized by a decrease in the slope of shear stress-shear strain curve at increasing values of shear stresses. Accordingly an increase in oil recovery is to be expected. This is seen in Figure 5.4.

5.6 EFFECT OF INFLOW PLANE PRESSURE

The effect of enhanced inflow plane pressure is presented in this section. A higher pressure value of 3.63 MPa instead of 1.79 Mpa, that is used throughout this analysis, is used on the inflow plane. Formation temperature used is 30°C and fluid injection is in the presence of surfactant. Newtonian behaviour of oil is

considered here. With a 4m long domain the saturation profile shows non-physical oscillations and after 20 minutes it breaks down. This may be attributed to the proximity of the outflow plane that precludes the use of a Dirichlet boundary condition. As discussed in Chapter 2, the use of Dirichlet boundary condition on the outflow plane is justified only if the saturation profile in the vicinity of the outflow plane is not greatly disturbed. When the inflow plane pressure is raised substantially the water-oil interface moves at a greater speed and reaches the outflow plane well before three hours. Therefore the saturation profile breaks down soon since the use of Dirichlet boundary condition is not proper. An interface of two immiscible fluids like oil and water is inherently unstable. Therefore with increasing inflow plane pressure the interface, which as such is unstable, becomes more prone to a collapse.

With the same enhanced inflow plane pressure, with a 8m long domain the saturation profile is continuous even after three hours. This simulation uses the domain decomposition code with two 4m long sub-domains. Results are shown in Table 5.4.

Table 5.1 : Properties Used in Numerical Calculation

Absolute Permeability, K = 132 Darcies

Porosity, ε = 0.375

Compressibility: oil, $\xi_o = 0.03447\ Pa^{-1}$, water, $\xi_w = 0.02137\ Pa^{-1}$

In-situ formation pressure = 1.31 MPa

Injection pressure = 1.793 MPa

Dynamic Viscosity (Pa-s)

T, ^oC	20	40	60	80
Oil	4.12	2.37	0.61	0.10
Water	1.03×10^{-3}	6.6×10^{-4}	4.73×10^{-4}	3.67×10^{-4}

Comparison showing the effect of grid refinement on saturation values in the oil recovery problem (Newtonian behaviour of oil) in the presence of surfactant.. Formation temperature is 40°C.

A: $\Delta x = 0.05m$; B: $\Delta x = 0.033m$

Comparison of saturation values.

x ⇒		0.5	1	1.5	2.0	2.5	3.0	3.5	4.0
t= 1.0	A	.3164	.2435	.2156	.2040	.2000	.2000	.2000	.2000
	B	.3146	.2461	.2084	.2051	.2000	.2000	.2000	.2000
t= 3.0	A	.4015	.3376	.2786	.2520	.2212	.2110	.2000	.2000
	B	.4021	.3382	.2772	.2516	.2219	.2106	.2000	.2000

Comparison of ppv values.

t ⇒	1	2	3
A	4.0616	7.8607	11.6167
B	4.0594	7.7132	11.4211

Table 5.3: Comparison of the solutions of the non-linear heat conduction problem (chapter 5) obtained by using direct matrix inverter and Gauss-Seidel iterations.

A: direct matrix inverter

B: Gauss-Seidel iterations.

All quantities are dimensionless.

x →		0.2	0.4	0.6	0.8	1.0	2.0	4.0
t=.18	A	.6634	.3872	.1983	.0893	.0356	.0001	0.0
	B	.6633	.3872	.1974	.0864	.0355	.0001	0.0
t=0.5	A	.8146	.6395	.4830	.3504	.2439	.0209	0.0
	B	.8146	.6400	.4831	.3504	.2438	.0209	0.0

Table 5.4: Results of enhanced inflow plane pressure simulation of 8m long domain with two 4m long sub-domains. Duration of the simulation is three hours and fluid injection is in the presence of surfactant. Oil is taken to be Newtonian. Formation temperature is 30°C.

Inflow plane pressure: 3.63 MPa

x in meters and t in hours.

x →	1	2	3	4	5	6	7	8
S_w	0.345	0.272	0.265	0.241	0.212	0.202	0.200	0.200

t →	0.5	1.0	1.5	2.0	2.5	3
ppv	2.217	3.770	5.310	7.001	8.752	10.422

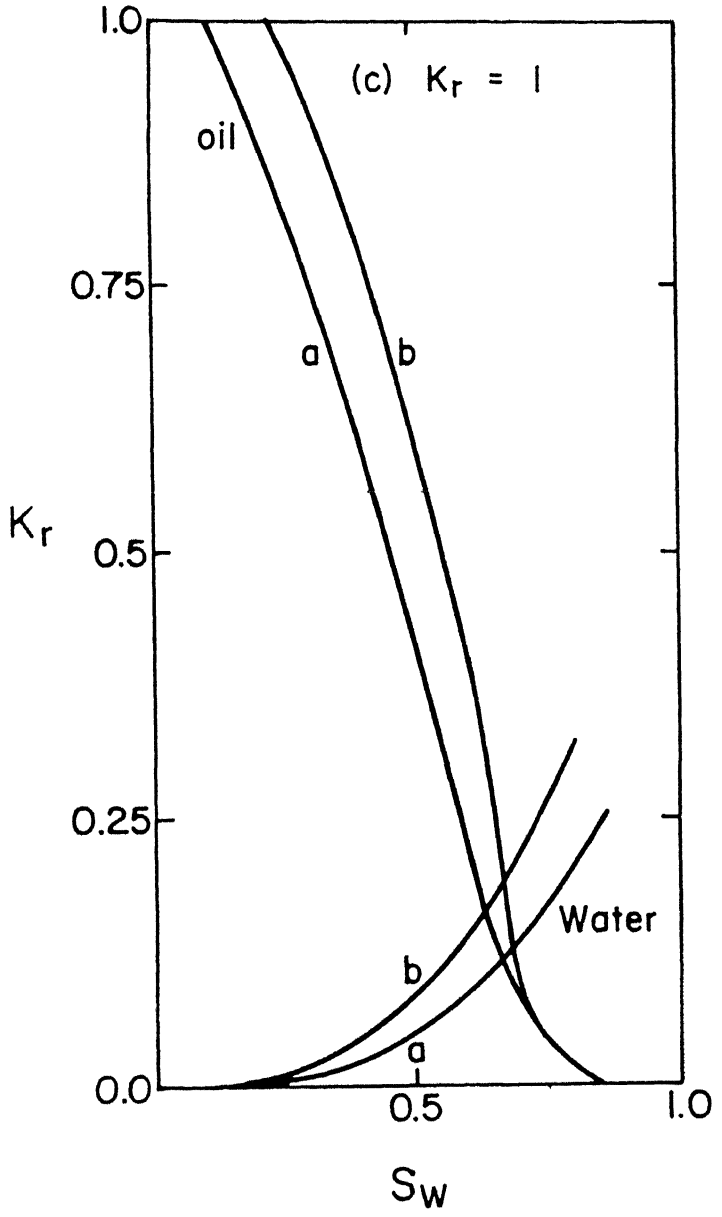


Figure 5.1a Relative permeability as function of saturation.

a) Without surfactant b) With surfactant c) Ideal case

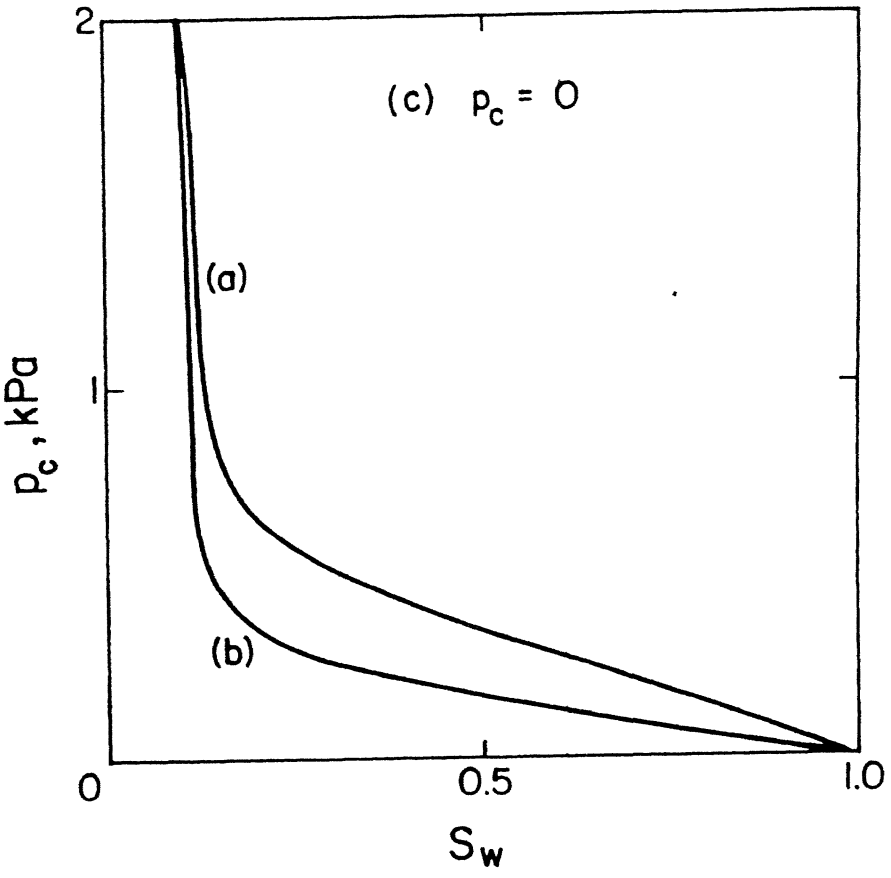


Figure 5.1b Capillary pressure as function of saturation.

a) Without surfactant b) With surfactant c) Ideal case

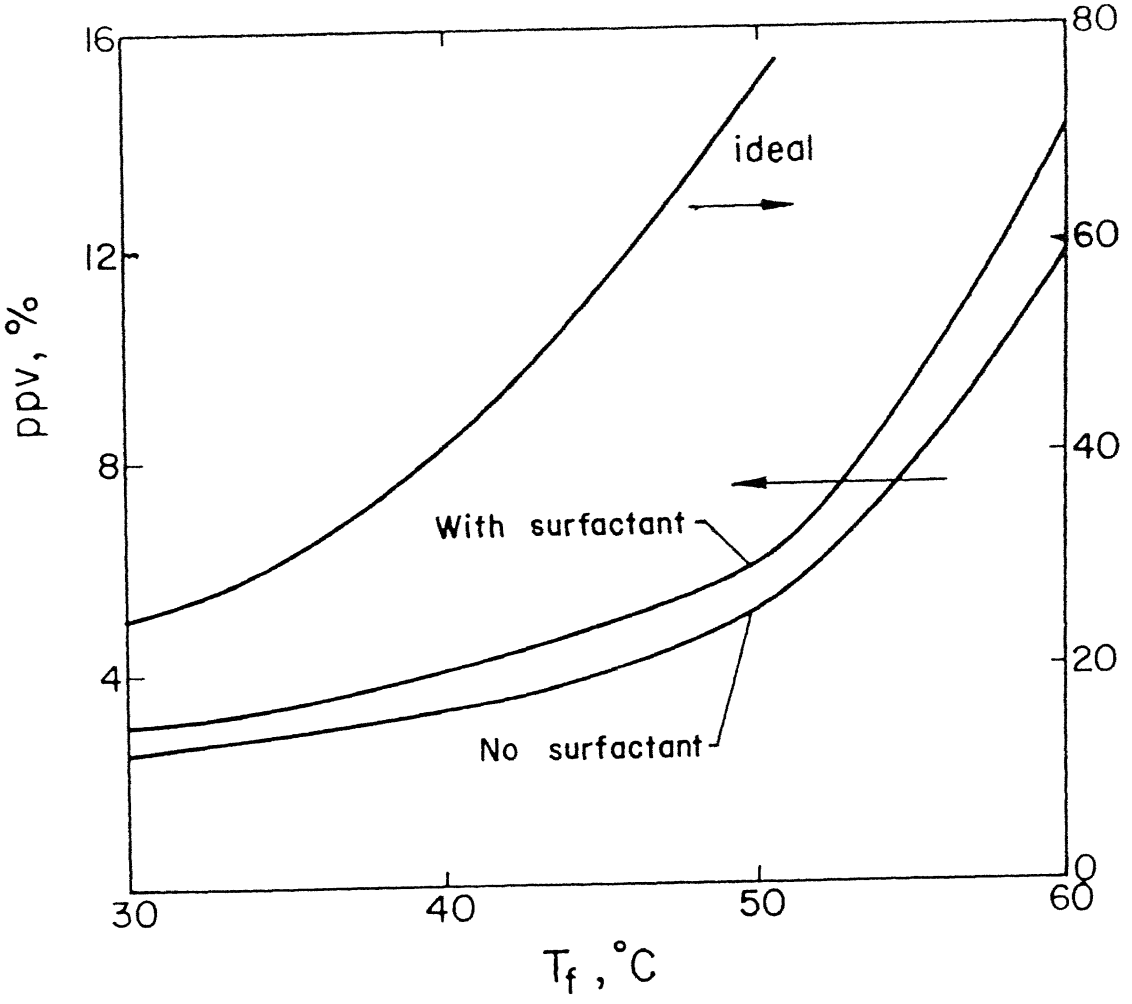


Figure 5.2 Oil recovery as a function of formation temperature.

a) Without surfactant b) With surfactant c) Ideal case

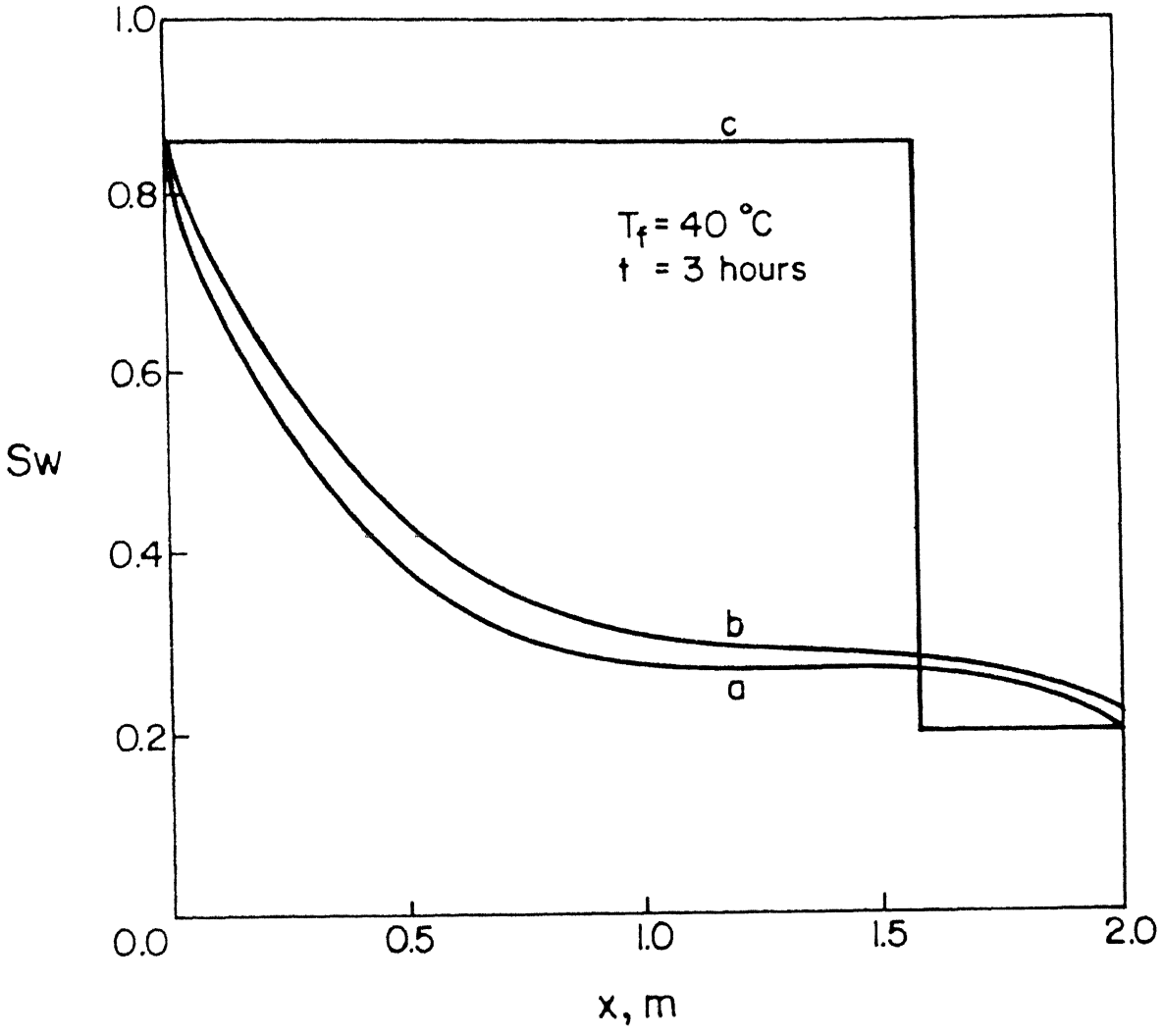


Figure 5.3 Variation of water saturation with distance at the end of three hours. a) Without surfactant b) With surfactant c) Ideal case

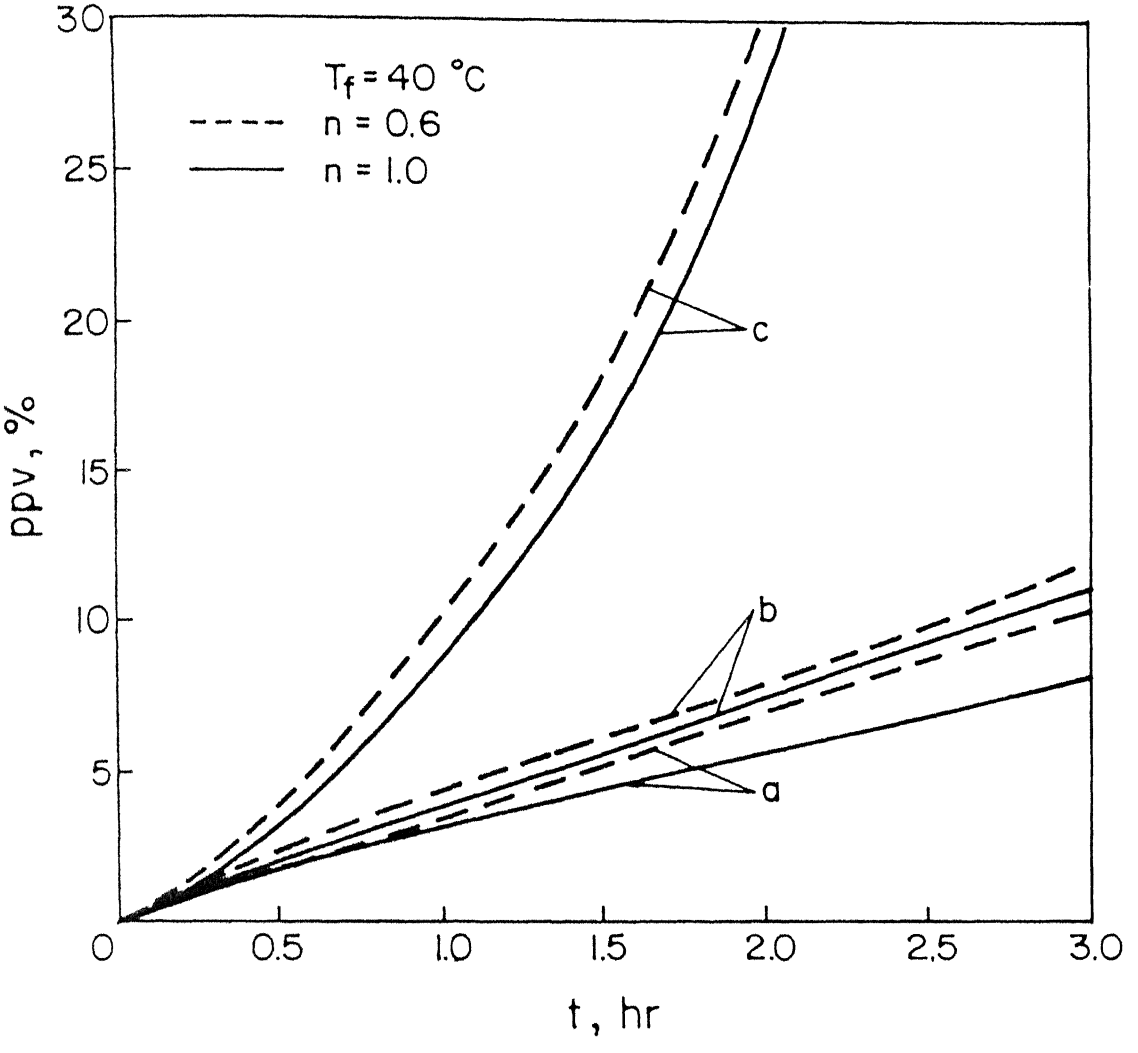


Figure 5.4 Effect of shear-thinning behaviour of oil. a) Without surfactant b) With surfactant c) Ideal case

The following extension to the present work is recommended to make numerical simulation of oil recovery complete and realistic.

1) Radial simulation:

In this work we have considereda laboratory scale rectangular domain with one injection well and one production well. In reality, reservoirs are characterized by one centrally located injection well radially sorrounded by production wells. Analysis of these reservoirs involve the use of radial coordinates. Instead of performing the analysis for the whole system, which may be very time consuming, a novel approach would be is to assume symmetry and analyze only one pair of injection well and production well and extend the results to the whole domain. In the vicinity of the wells near-field effects are predominant and they should be tackled carefully. Domain decomposition finds utility here as the affected regions can be localized by assigning separate sub-domains to them and can be studied closely.

2) Stabilty limits of simulation:

The numerical simulation presented in this work has been done for one particular formation i.e. one particular set of formation and surfactant properties. Owing to high degree of non-linearity involved in the governing reservoir equations, applicability of this simulation code to other sets of properties cannot be guaranteed. Hence to ascertain versatility of the present formulation and computer code, simulation should be performed with other formation propereties as well. There is considerable evidence that with certain properties the water saturation profiles become unstable. This leads to mixing of oil and water and a fall in oil recovery with increasing injection pressure. This is a practical problem that deserves attention.

3) Three-phase flow:

Besides oil and water, the third phase that is present in certain reservoirs is air, though comparatively much less in quantity. A three phase simulation (oil, water and air) would thus lend more credibility to this work. An oil-water-air-steam simulation,

though difficult must also be taken up for analysis.

4) Adsorption-Desorption in the formation:

In the case of surfactant flooding a considerable amount of surfactant is lost to the formation because of adsorption. Desorption i.e. flow of surfactant from formation to the surfactant solution also takes place, though much less in magnitude. Bacause of these mass transfer phenomena the constututive relations must be modified in conjunction with the variation of concentration of surfactant solution. An additional mass transfer equation comes into picture and it must be solved along with the pressure equations.

5) Three dimensional simulation:

An extension of the present is a three dimensional simulation including inhomogeneity and anisotropy of the formation. This will make reservoir simulation computationally more expensive. Domain decomposition with parallelization developed in the present work is the best option to tackle this problem.

6) Parallelization:

Implementation of the parallel code developed here on a parallel computer remains incomplete. It could not be done because of lack of availibility of parallel computing facility at IIT Kanpur. This implementation would not only save a lot of computational time but it would also enable us to perform simulation for real life reservoirs.

7) Interface treatment:

Uzawa's algorithm that is used here is one of the rudimentary techniques for interface treatment in domain deconposition. Many advanced parallelizable techniques are now available (*Glowinski (1983)*). Use of these techniques can be expected to enhance the convergence rate of the interface iterations and make the whole simulation faster. These techniques can be tested systematically in the future.

1) Arthur G. S. Pillet, *Heat Transfer During Hot Fluid Injection Into an Oil Reservoir*, Heavy Oil Seminar, 1965

2) Aziz K. , *Modeling of Thermal Oil Recovery Processes*, In Mathematical and Computational Methods in Seismic Exploration and Reservoir Modeling, SIAM, Philadelphia, 1986

3) Barenblatt G. I. , Entov V. M. , Ryzhik V. M. , *Theory of Fluid Flow Through Natural Rocks*, Klower Academic Publishers, 1990

4) Boberg T. C. , *Thermal Methods of Oil Recovery*, Exxon Monograph, John Wiley, 1988

5) Duff I. S. , *MA28 - A Set of Fortran Subroutines For Sparse Unsymmetric Linear Equations*, AERE Harwell, UK, 1980

6) Ewing R. E. , *Mathematics of Reservoir Simulation*, SIAM Publications on Applied Mathematics, 1983

7) Ganagi M. S. , Singh K. P. ,Athithan G. and Atre M. V. , *A Fluid-Dynamics Study on ANURAG'S Parallel Computer, PACE-8*, Current Science, Vol 60, 1991

8) Glowinski R. , Dinh Q. V. and Periaux J. , *Domain Decomposition Methods For Non-Linear Problems in Fluid Dynamics*, Comp. Methods in Applied Mechanics and Engineering, Vol 40, 1983

9) Jim Douglas Jr. , *Finite Difference Methods for Two-Phase Incompressible Flow in Porous Media*, SIAM J. Num. Analysis, Vol 20, 1983

10) Shah D. O. and Schechter R. S. , *Improved Oil Recovery by Surfactant and Polymer Flooding*, Academic Press, 1977

11) Yagawa G. , Soneda N. and Yoshimura S. , *A Large Scale Finite Element Analysis Using Domain Decomposition Method on Parallel Computer*, Computers and Structures, Vol 38, 1991

APPENDIX A1: ITERATIVE TECHNIQUES FOR NON-LINEAR EQUATIONS

In this section we describe two iterative schemes, Picard iterations and Newton-Raphson iterations, that have been used in this study to solve non-linear algebraic equations.

A1.1 PICARD ITERATIONS

Let us first consider the single variable problem. The equation to be solved is

$$F(x) = 0 \tag{1}$$

where F(x) is a non-linear function of the variable x. In this technique, we first recast Equation 1 in the following form

$$x = f(x) \tag{2}$$

This recasting can be done in many ways and there is no uniqueness in this step. One then assumes a starting (or first iterate) value, x^1, for the variable x and uses this in f(x) of Equation 2 to get a new and hopefully, improved estimate, x^2. Thus

$$x^2 = f(x^1) \tag{3}$$

x^2 is now used as the argument of f(x) in Equation 2 to yield the next estimate, x^3. This procedure is continued and we have the algorithm as

$$x^{k+1} = f(x^k) \; ; \; k = 1, 2, \ldots \tag{4}$$

Computations are stopped when the successive iterates are such that

$$|x^{k+1} - x^k| \leq \varepsilon \tag{5a}$$

and/or

$$|F(x^{k+1})| \leq \varepsilon \tag{5b}$$

where ε is the desired tolerance.

This technique can now be easily extended to the more general case involving N variables, $x_1, x_2, \ldots, x_N$. The i^{th} equation, $F_i(x) = 0$, in Equation 1 can be rearranged as

$$x_i = f_i(x) \; ; \; i = 1, 2, \ldots, N \tag{6}$$

In the matrix form it may be written as

$$x = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_N \end{bmatrix} = f(x) = \begin{bmatrix} f_1(x) \\ f_2(x) \\ \vdots \\ f_N(x) \end{bmatrix} \tag{7}$$

In analogy with Equation 4, the algorithm can be writen as

$$x_1^{k+1} = f_1[x^k] = f_1[x_1^k, x_2^k, \ldots, x_N^k] \qquad (8)$$

One starts with a choice , x^1, of N variables and iterates until convergence criteria similar to Equations 5a and 5b are satisfied on all x_i and $F_i[x]$.

A1.2 NEWTON - RAPHSON ITERATIONS

A very popular method for solving sets of non-linear algebraic equations is the Newton-Raphson technique.

For the single variable case, we have

$$F(x) = 0 \qquad (10)$$

As described before, x^k is the estimate of the root, α, at the k^{th} iteration and x^{k+1} is the improved estimate at the next iteration, then the Newton-Raphson algorithm is

$$x^{k+1} = x^k - \frac{F(x^k)}{F'(x^k)} \qquad (11)$$

where $F'(x^k)$ ($= \frac{dF}{dx}$) is evaluated at x^k. The convergence criterion of the Newton-Raphson technique is that if the function, F(x), has a continuous derivative in the neighborhood of a root, α, and if the derivative, $F'(x)$, is non-zero in the neighbohood of this root, then the Newton-Raphson technique converges to α, provided x^1 (the first guess) is taken sufficiently close this root. Computations are stopped when the conditions in Equations 5a and 5b are met.

The extension of this technique to a set of N equations in N variables, $x_1, x_2, \ldots, x_N$, is straightforward. The i^{th} equation from the set, Equation 10, is

$$F_i(x) = 0 = F_i(x_1, x_2, \ldots, x_N) \; ; \; i = 1, 2, \ldots, N \qquad (12)$$

Iterations are then governed by

$$-\begin{bmatrix} F_1[x^k] \\ F_2[x^k] \\ \vdots \\ F_N[x^k] \end{bmatrix}_{N\times 1} = \begin{bmatrix} \partial F_1/\partial x_1 & \partial F_1/\partial x_2 & \cdots & \partial F_1/\partial x_N \\ \partial F_2/\partial x_1 & \cdots & \cdots & \cdots \\ \vdots & \cdots & \cdots & \vdots \\ \partial F_N/\partial x_1 & \partial F_N/\partial x_2 & \cdots & \partial F_N/\partial x_N \end{bmatrix}_{N\times N} \begin{bmatrix} x_1^{k+1} - x_1^k \\ x_2^{k+1} - x_2^k \\ \vdots \\ x_N^{k+1} - x_N^k \end{bmatrix}_{N\times 1} \qquad (13)$$

or $$-F[x^k] = A[x^k]\,[x^{k+1} - x^k] \qquad (14)$$

where $A[x^k]$ is the Jacobian matrix evaluated at x^k and it is a matrix of constants. Equation 14 can be rewritten as

$$x^{k+1} = x^k - \left[A[x^k] \right]^{-1} F[x^k] \qquad (15)$$

It can be easily seen that Equation 11 is a special case of the more general Equation 15. Iterations stop when the convergence criteria as outlined in Equations 5a and 5b are satisfied for all x_i and $F_i[x]$.

A1.3 APPLICATION OF NEWTON - RAPHSON TECHNIQUE

We now use the Newton-Raphson technique to solve the two dimensional non-linear transient heat conducton problem.

$$\underset{\sim}{\nabla}.\ K\ \underset{\sim}{\nabla}\ T = \frac{\partial T}{\partial t} \qquad (16)$$

with the initial and boundary conditions,

$$\text{at } t=0 \qquad T = T_o \qquad (17)$$

$$\text{at } t>0, \quad \text{at } x = 0 \qquad T = T_1 \qquad (18)$$

$$\text{at } x = X \qquad T = T_2 \qquad (19)$$

$$\text{at } y = 0 \qquad \frac{\partial T}{\partial y} = 0 \qquad (20)$$

$$\text{at } y = Y \qquad \frac{\partial T}{\partial y} = 0 \qquad (21)$$

In Equation 16, K is the thermal conductivity and its variation with temperature is given as

$$K = 1 - \beta T \qquad (22)$$

Equations 16-21 are discretized using central differencing with implicit time stepping. The control volume is shown in Figure A1.1.

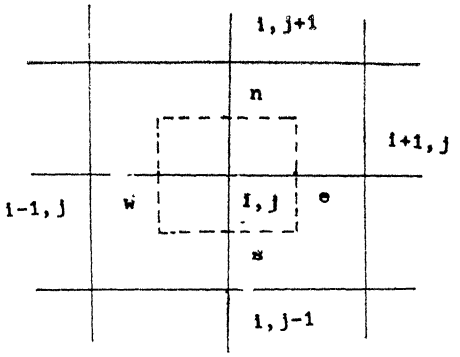


Figure A1.1

i and j are the grid locations in x and y directions respectively. N is the number of nodes in the x-direction and M that in the y-direction. Δx and Δy define a Cartesian mesh and Δt is the time step. (n+1) indicates the current time step and n the previous time step. Upon discretization, we get

For the interior points; i = 2, N-1 and j =2, M-1

$$-T_{i,j}^{n+1}\left[\frac{K_e}{\Delta x^2}+\frac{K_w}{\Delta x^2}+\frac{K_s}{\Delta y^2}+\frac{K_n}{\Delta y^2}\right]^{n+1}+T_{i+1,j}^{n+1}\left(\frac{K_e}{\Delta x^2}\right)^{n+1}$$

$$+T_{i-1,j}^{n+1}\left(\frac{K_w}{\Delta x^2}\right)^{n+1}+T_{i,j+1}^{n+1}\left(\frac{K_n}{\Delta y^2}\right)^{n+1}+T_{i,j-1}^{n+1}\left(\frac{K_s}{\Delta y^2}\right)^{n+1}$$

$$+\frac{T_{i,j}^{n}}{\Delta t}=0 \qquad (23)$$

and the discretized boundary conditions are:

$$\text{for } i = 1, \quad T_{i,j} = T_1 \qquad (24)$$

$$\text{for } i = N, \quad T_{i,j} = T_2 \qquad (25)$$

$$\text{for } j = 1, \quad T_{i,j+1} - T_{i,j} = 0 \qquad (26)$$

$$\text{for } j = M, \quad T_{i,j} - T_{i,j-1} = 0 \qquad (27)$$

K_e, K_w, K_n, K_s are the harmonic averages of the values of K at nodes that lie on either side. Thus, by using finite differences we have reduced the governing partial differential equation and the boundary conditions, Equations 16-21, to a system of NM non-linear algebraic equations, Equations 23-27, which may be written as

$$F_{i,j}[\,T\,] = 0 = F_{i,j}[\,T_{11}, T_{12}, \ldots, T_{N,M-1}, T_{NM}] \qquad (28)$$

where the NM unknowns are the temperatures at the NM grid points. For simplicity, we reduce the two dependent subscripts, i and j, to one subscript, j + (i-1)M, where i and j hold the same meaning. Equation 28 may now be written as

$$F_{j+(i-1)M}[\,T\,] = 0 \qquad (29)$$

where the NM unknown temperatures are now $T_{j+(i-1)M}$ for all i and j.

The system of equations, Equations 23-27, is now solved using the Newton-Raphson technique. The first step would be is to form the Jacobian matrix A. A close look at Equations 23-27 shows that

any $F_{i,j}$ has only the temperature at the grid point i,j and the temperatures at the four neighbouring grid points as its variables. Therefore, we evaluate only the derivatives with respect to these temperatures and safely put other derivatives to zero. The non-zero derivatives are

$$1)\quad \left.\frac{\partial F_{j+(i-1)M}}{\partial T}\right|_{T_{i,j}} \tag{30}$$

For i = 2, N-1 and j = 2, M-1

$$= -\left[\frac{k_e}{\Delta x^2}+\frac{k_w}{\Delta x^2}+\frac{k_s}{\Delta y^2}+\frac{k_n}{\Delta y^2}+\frac{1}{\Delta t}\right]-T_{i,j}\left[\frac{1}{\Delta x^2}\frac{\partial k_e}{\partial T_{i,j}}+\right.$$

$$\left.\frac{1}{\Delta x^2}\frac{\partial k_w}{\partial T_{i,j}}+\frac{1}{\Delta y^2}\frac{\partial k_n}{\partial T_{i,j}}+\frac{1}{\Delta y^2}\frac{\partial k_s}{\partial T_{i,j}}\right]+\frac{T_{i+1,j}}{\Delta x^2}\frac{\partial k_e}{\partial T_{i,j}}+\frac{T_{i-1,j}}{\Delta x^2}\frac{\partial k_w}{\partial T_{i,j}}$$

$$+\frac{T_{i,j+1}}{\Delta y^2}\frac{\partial k_n}{\partial T_{i,j}}+\frac{T_{i,j-1}}{\Delta y^2}\frac{\partial k_s}{\partial T_{i,j}}$$

For i = 1 = 1
For i = N = 1
For j = 1 = -1
For j = M = 1

$$2)\quad \left.\frac{\partial F_{j+(i-1)M}}{\partial T}\right|_{T_{i+1,j}} \tag{31}$$

For i = 2, N-1 and j = 2, M-1

$$= -\frac{T_{i,j}}{\Delta x^2}\frac{\partial k_e}{\partial T_{i+1,j}}+\frac{k_e}{\Delta x^2}+\frac{T_{i+1,j}}{\Delta x^2}\frac{\partial k_e}{\partial T_{i+1,j}}$$

For i = 1 = 0
For i = N = 0
For j = 1 = 0
For j = M = 0

$$3)\ \left.\frac{\partial F_{j+(i-1)M}}{\partial T}\right|_{T_{i-1,j}} \qquad (32)$$

For i = 2, N-1 and j = 2, M-1

$$= -\frac{T_{i,j}}{\Delta x^2}\frac{\partial k_w}{\partial T_{i-1,j}} + \frac{k_w}{\Delta x^2} + \frac{T_{i-1,j}}{\Delta x^2}\frac{\partial k_w}{\partial T_{i-1,j}}$$

For i = 1 = 0

For i = N = 0

For j = 1 = 0

For j = M = 0

$$4)\ \left.\frac{\partial F_{j+(i-1)M}}{\partial T}\right|_{T_{i,j+1}} \qquad (33)$$

For i = 2, N-1 and j = 2, M-1

$$= -\frac{T_{i,j}}{\Delta y^2}\frac{\partial k_n}{\partial T_{i,j+1}} + \frac{k_n}{\Delta y^2} + \frac{T_{i,j+1}}{\Delta y^2}\frac{\partial k_n}{\partial T_{i,j+1}}$$

For i = 1 = 0

For i = N = 0

For j = 1 = 1

For j = M = 0

$$5)\ \left.\frac{\partial F_{j+(i-1)M}}{\partial T}\right|_{T_{i,j-1}} \qquad (34)$$

For i = 2, N-1 and j = 2, M-1

$$= -\frac{T_{i,j}}{\Delta y^2}\frac{\partial k_s}{\partial T_{i,j-1}} + \frac{k_s}{\Delta y^2} + \frac{T_{i,j-1}}{\Delta y^2}\frac{\partial k_s}{\partial T_{i,j-1}}$$

For i = 1 = 0

For i = N = 0

For j = 1 = 1

For j = M = 0

In all the derivatives calculated above, derivatives of K_i (i=e,w,n,s) with respect to T are calculated along the same lines as the example given below.

$$\frac{\partial K_e}{\partial T_{i+1,j}} = -\beta/2 \quad \text{and} \quad \frac{\partial K_e}{\partial T_{i-1,j}} = 0 \tag{35}$$

With the derivatives calculated, the Jacobian matrix A is formed as described in Equation 13. We now outline the algorithmic steps to solve this transient problem.

1) For every time step choose an initial guess (first iterate) for the temperatures. This initial guess may be readily chosen as the temperature values at the previous time step.

2) With the initial guess chosen, iterate the temperatures within the time step using the Newton-Raphson scheme, Equation 15,

$$[\ T\ ]^{k+1} = [\ T\ ]^{k} - \left[\ A\ [\ T^{k}]\ \right] \quad F\ [\ T^{k}] \tag{36}$$

where A [T^k] is the Jacobian calculated at [T]^k. Similarly F [T^k] are the functions, $F_{j+(i-1)M}$, calculated at [T]^k.

3) Continue the iterations until convergence is achieved. The convergence criterion can be written as,

$$\left|[\ T\ ]^{k+1} - [\ T\ ]^{k}\right| \leq \varepsilon \tag{37}$$

where ε is the desired tolerence.

4) Upon convergence proceed to next time step and go through the same sequence.

APPENDIX A2: MATRIX STRUCTURE FOR OIL RECOVERY PROBLEM

In this section we list the values of coefficients of the discretized pressure equations, Equations 4.2 and 4.3. These coefficients are the non-zero entries of the matrix described in Chapter 4 and shown in Figure 4.3.

A2.1 PICARD ITERATIONS

On using Picard iterations, the coefficients are as follows:

Here $\theta = \dfrac{k\, k_{ro} \rho_o}{\varepsilon\, \mu_o}$ and $\gamma = \dfrac{k\, k_{rw} \rho_w}{\varepsilon\, \mu_w}$

$$A_o = -\left\{\frac{\theta_e}{\rho_o\,(\Delta x)^2}\right\}^{n+1}_{i,j},$$

$$B_o = \left[\frac{S_o \xi_o - \left(\frac{dS_w}{dP_c}\right)}{\Delta t} + \frac{1}{\rho_o}\left\{\frac{\theta_e + \theta_w}{(\Delta x)^2} + \frac{\theta_n + \theta_s}{(\Delta y)^2}\right\}_{i,j}\right]^{n+1},$$

$$C_o = \left\{\frac{dS_w}{dP_c}\right\}^{n+1}_{i,j} \frac{1}{\Delta t}, \quad D_o = -\left\{\frac{\theta_w}{\rho_o (\Delta x)^2}\right\}^{n+1}_{i,j},$$

$$F_o = -\left\{\frac{\theta_s}{\rho_o\,(\Delta y)^2}\right\}^{n+1}_{i,j},$$

$$G_o = \left[\left\{\frac{S_o \xi_o - \frac{dS_w}{dP_c}}{\Delta t}\right\}^{n+1} \cdot P_o^n + \frac{1}{\Delta t}\left\{\frac{dS_w}{dP_c}\right\}^{n+1} P_w^n\right]_{i,j}$$

$$A_w = -\left\{\frac{\gamma_e}{\rho_w\,(\Delta x)^2}\right\}^{n+1}_{i,j},$$

$$B_w = \left[\frac{S_w \xi_w - \left(\frac{dS_w}{dP_c}\right)}{\Delta t} + \frac{1}{\rho_w}\left\{\frac{\gamma_e + \gamma_w}{(\Delta x)^2} + \frac{\gamma_n + \gamma_s}{(\Delta y)^2}\right\}_{i,j}\right]^{n+1},$$

Other derivatives of θ and γ are calculated similarly. Here

$$a_o = \left(S_o \xi_o - \frac{dS_w}{dP_c} \right)_{i,j} \quad ; \qquad a_w = \left(S_w \xi_w - \frac{dS_w}{dP_c} \right)_{i,j}$$

$$\text{and } b_o = b_w = \left. \frac{dS_w}{dP_c} \right|_{i,j}$$

$$A_o = - \frac{P_{oi,j}}{\rho_o \Delta x^2} \left[\frac{\partial \theta_e}{\partial P_{oi+1,j}} \right] + \frac{\theta_e}{\Delta x^2 \rho_o} + \frac{P_{oi+1,j}}{\rho_o \Delta x^2} \left[\frac{\partial \theta_e}{\partial P_{oi+1,j}} \right]$$

$$D_o = - \frac{P_{oi,j}}{\rho_o \Delta x^2} \left[\frac{\partial \theta_w}{\partial P_{oi-1,j}} \right] + \frac{\theta_w}{\Delta x^2 \rho_o} + \frac{P_{oi-1,j}}{\rho_o \Delta x^2} \left[\frac{\partial \theta_w}{\partial P_{oi-1,j}} \right]$$

$$E_o = - \frac{P_{oi,j}}{\rho_o \Delta y^2} \left[\frac{\partial \theta_n}{\partial P_{oi,j+1}} \right] + \frac{\theta_n}{\Delta y^2 \rho_o} + \frac{P_{oi,j+1}}{\rho_o \Delta y^2} \left[\frac{\partial \theta_n}{\partial P_{oi,j+1}} \right]$$

$$F_o = - \frac{P_{oi,j}}{\rho_o \Delta y^2} \left[\frac{\partial \theta_s}{\partial P_{oi,j-1}} \right] + \frac{\theta_s}{\Delta y^2 \rho_o} + \frac{P_{oi,j-1}}{\rho_o \Delta y^2} \left[\frac{\partial \theta_s}{\partial P_{oi,j-1}} \right]$$

$$C_o = - \frac{P_{oi,j}}{\rho_o} \left[\frac{1}{\Delta x^2} \frac{\partial \theta_e}{\partial P_{wi,j}} + \frac{1}{\Delta x^2} \frac{\partial \theta_w}{\partial P_{wi,j}} + \frac{1}{\Delta y^2} \frac{\partial \theta_n}{\partial P_{wi,j}} + \frac{1}{\Delta y^2} \frac{\partial \theta_s}{\partial P_{wi,j}} + \frac{\rho_o}{\Delta t} \frac{\partial a_o}{\partial P_{wi,j}} \right] - \left[\frac{b_o}{\Delta t} + \frac{P_{wi,j}}{\Delta t} \frac{\partial b_o}{\partial P_{wi,j}} \right] + \frac{P_{oi+1,j}}{\Delta x^2 \rho_o} \frac{\partial \theta_e}{\partial P_{wi,j}} + \frac{P_{oi-1,j}}{\Delta x^2 \rho_o} \frac{\partial \theta_w}{\partial P_{wi,j}} + \frac{P_{oi,j+1}}{\Delta y^2 \rho_o} \frac{\partial \theta_n}{\partial P_{wi,j}} + \frac{P_{oi,j-1}}{\Delta y^2 \rho_o} \frac{\partial \theta_s}{\partial P_{wi,j}} + \frac{P^n_{oi,j}}{\Delta t} \frac{\partial a_o}{\partial P_{wi,j}} + \frac{P^n_{wi,j}}{\Delta t} \frac{\partial b_o}{\partial P_{wi,j}}$$

$$B_o = - \frac{P_{oi,j}}{\rho_o} \left[\frac{1}{\Delta x^2} \frac{\partial \theta_e}{\partial P_{oi,j}} + \frac{1}{\Delta x^2} \frac{\partial \theta_w}{\partial P_{oi,j}} + \frac{1}{\Delta y^2} \frac{\partial \theta_n}{\partial P_{oi,j}} + \frac{1}{\Delta y^2} \frac{\partial \theta_s}{\partial P_{oi,j}} + \frac{\rho_o}{\Delta t} \frac{\partial a_o}{\partial P_{oi,j}} \right] - \frac{1}{\rho_o} \left[\frac{\theta_e}{\Delta x^2} + \frac{\theta_w}{\Delta x^2} + \frac{\theta_n}{\Delta y^2} + \frac{\theta_s}{\Delta y^2} + \frac{a_o \rho_o}{\Delta t} \right] - \frac{P_{wi,j}}{\Delta t} \frac{\partial b_o}{\partial P_{oi,j}} + \frac{P_{oi+1,j}}{\Delta x^2 \rho_o} \frac{\partial \theta_e}{\partial P_{oi,j}} + \frac{P_{oi-1,j}}{\Delta x^2 \rho_o} \frac{\partial \theta_w}{\partial P_{oi,j}} + \frac{P_{oi,j+1}}{\Delta y^2 \rho_o} \frac{\partial \theta_n}{\partial P_{oi,j}} + \frac{P_{oi,j-1}}{\Delta y^2 \rho_o} \frac{\partial \theta_s}{\partial P_{oi,j}} + \frac{P^n_{oi,j}}{\Delta t} \frac{\partial a_o}{\partial P_{oi,j}} + \frac{P^n_{wi,j}}{\Delta t} \frac{\partial b_o}{\partial P_{oi,j}}$$

$$A_w = -\frac{P_{wi,j}}{\rho_w \Delta x^2}\left[\frac{\partial \gamma_e}{\partial P_{wi+1,j}}\right] + \frac{\gamma_e}{\Delta x^2 \rho_w} + \frac{P_{wi+1,j}}{\rho_w \Delta x^2}\left[\frac{\partial \gamma_e}{\partial P_{wi+1,j}}\right]$$

$$D_w = -\frac{P_{wi,j}}{\rho_w \Delta x^2}\left[\frac{\partial \gamma_w}{\partial P_{wi-1,j}}\right] + \frac{\gamma_w}{\Delta x^2 \rho_w} + \frac{P_{wi-1,j}}{\rho_w \Delta x^2}\left[\frac{\partial \gamma_w}{\partial P_{wi-1,j}}\right]$$

$$E_w = -\frac{P_{wi,j}}{\rho_w \Delta y^2}\left[\frac{\partial \gamma_n}{\partial P_{wi,j+1}}\right] + \frac{\gamma_n}{\Delta y^2 \rho_w} + \frac{P_{wi,j+1}}{\rho_w \Delta y^2}\left[\frac{\partial \gamma_n}{\partial P_{wi,j+1}}\right]$$

$$F_w = -\frac{P_{wi,j}}{\rho_w \Delta y^2}\left[\frac{\partial \gamma_s}{\partial P_{wi,j-1}}\right] + \frac{\gamma_s}{\Delta y^2 \rho_w} + \frac{P_{wi,j-1}}{\rho_w \Delta y^2}\left[\frac{\partial \gamma_s}{\partial P_{wi,j-1}}\right]$$

$$C_w = -\frac{P_{wi,j}}{\rho_w}\left[\frac{1}{\Delta x^2}\frac{\partial \gamma_e}{\partial P_{oi,j}} + \frac{1}{\Delta x^2}\frac{\partial \gamma_w}{\partial P_{oi,j}} + \frac{1}{\Delta y^2}\frac{\partial \gamma_n}{\partial P_{wi,j}} + \frac{1}{\Delta y^2}\frac{\partial \gamma_s}{\partial P_{oi,j}} + \frac{\rho_w}{\Delta t}\frac{\partial a_w}{\partial P_{oi,j}}\right] - \left[\frac{b_w}{\Delta t} + \frac{P_{oi,j}}{\Delta t}\frac{\partial b_w}{\partial P_{oi,j}}\right] + \frac{P_{wi+1,j}}{\Delta x^2 \rho_w}\frac{\partial \gamma_e}{\partial P_{oi,j}} + \frac{P_{wi-1,j}}{\Delta x^2 \rho_w}\frac{\partial \gamma_w}{\partial P_{oi,j}} + \frac{P_{wi,j+1}}{\Delta y^2 \rho_w}\frac{\partial \gamma_n}{\partial P_{oi,j}} + \frac{P_{wi,j-1}}{\Delta y^2 \rho_w}\frac{\partial \gamma_s}{\partial P_{oi,j}} + \frac{P^n_{wi,j}}{\Delta t}\frac{\partial a_w}{\partial P_{oi,j}} + \frac{P^n_{oi,j}}{\Delta t}\frac{\partial b_w}{\partial P_{oi,j}}$$

$$B_w = -\frac{P_{wi,j}}{\rho_w}\left[\frac{1}{\Delta x^2}\frac{\partial \gamma_e}{\partial P_{wi,j}} + \frac{1}{\Delta x^2}\frac{\partial \gamma_w}{\partial P_{wi,j}} + \frac{1}{\Delta y^2}\frac{\partial \gamma_n}{\partial P_{wi,j}} + \frac{1}{\Delta y^2}\frac{\partial \gamma_s}{\partial P_{wi,j}} + \frac{\rho_w}{\Delta t}\frac{\partial a_w}{\partial P_{wi,j}}\right] - \frac{1}{\rho_w}\left[\frac{\gamma_e}{\Delta x^2} + \frac{\gamma_w}{\Delta x^2} + \frac{\gamma_n}{\Delta y^2} + \frac{\gamma_s}{\Delta y^2} + \frac{a_w \rho_w}{\Delta t}\right] - \frac{P_{oi,j}}{\Delta t}\frac{\partial b_w}{\partial P_{wi,j}} + \frac{P_{wi+1,j}}{\Delta x^2 \rho_w}\frac{\partial \gamma_e}{\partial P_{wi,j}} + \frac{P_{wi-1,j}}{\Delta x^2 \rho_w}\frac{\partial \gamma_w}{\partial P_{wi,j}} + \frac{P_{wi,j+1}}{\Delta y^2 \rho_w}\frac{\partial \gamma_n}{\partial P_{wi,j}} + \frac{P_{wi,j-1}}{\Delta y^2 \rho_w}\frac{\partial \gamma_s}{\partial P_{wi,j}} + \frac{P^n_{wi,j}}{\Delta t}\frac{\partial a_w}{\partial P_{wi,j}} + \frac{P^n_{oi,j}}{\Delta t}\frac{\partial b_w}{\partial P_{wi,j}}$$